Xor

Xor, chi era costui? Bene raga, questo è il primo articolo della UIC e vorrei iniziare col parlarvi dello Xor che è la più comune forma di crittazione usata per i vari seriali e password, ma anche per nascondere da occhi indiscreti una stringa, l'articolo avrà un'impronta tecnica dal momento che le varie tecniche di Xoring vi verranno illustrate e spiegate alla meglio dal mio amico T3x (dico bene T3x? :).
C'mon baby, la parola Xor non è altro che l'abbreviazione del termine "Exclusive Or", in italiano "Or esclusivo", ma per capire come funziona dobbiamo parlare per prima cosa dell'OR. Se mentre debuggate vi doveste trovare davanti una cosa del genere:

xxxx:0040123456 OR eax, 64h ("h" stà per esadecimale)

e non sapete a cosa serva, ecco che vi vengo in aiuto io, poniamo il caso che in eax sia contenuto il valore 58h, cosa fa l'OR? Semplicissimo, per prima cosa converte i due numeri da esadecimali in binari cioè:

eax = 58h = 1011000
64h = 1100100

e poi ne fa un confronto con i criteri appunto.....dell'OR :)) per ora vi mostro il risultato e poi ve lo spiego:

```
1011000  OR
1100100    =
 ----------
```

1111100 = 7c

il motivo per il quale è uscito 7c è che l'OR opera in questo modo:

A or B = X

```
 A B  |  X
0  0  |  0
 0  1  |  1
0  0  |  1
0  1  |  1
```

cioè, riporta una valore 0 (FALSO) solo se nel confronto tra due membri del numero trova due zeri, altrimenti, qualunque siano i valori (1 e 0, 0 e 1, 1 e 1) riporta 1 (cioè vero). Ma parliamo dello XOR, prendiamo i due numeri e convertiamoli in forma binaria:

eax = 58h = 1011000
64h = 1100100

li Xoriamo e divengono:

```
1011000   XOR
1100100     =
 ----------
```

0111100 = 3c

tutto ciò perchè lo XOR riporta un valore di 0 se due membri del numero sono uguali ed un valore di 1 se due membri sono diversi, comunque ecco la consueta tabella:

A xor B = X

```
A   B  |  X
0  0  |  0
 0  1  |  1
1  0  |  1
1  1  |  0
```

Vi starete chiedendo: ma come si xorano le password?
Bhè il meccanismo vero e proprio vi verrà spiegato da T3x, comunque il modo più semplice è quello di decidere una password, xorarla con un valore, poi prendere la password inserita, xorarla con lo stesso valore e vedere se combaciano, ecco un esempio in codice assembly:

----------8< Start code ----------

```
.286
.molel small
.stack 100h

.DATA
```

```
Password   db      04Ch                        ; poniamo il caso che questo sia il                                        ; numero 5 xorato con un valore
                                               ; di 20h (è sparato a caso)

Bravo       db      ' Bravo!!! ', 0Dh, 0Ah, '$'
Errato      db      ' Hai sbagliato!!! ', 0Dh, 0Ah, '$'

; supponiamo anche di avere in eax il numero inserito da voi, non scrivo il      codice perchè per il momento non vi interessa e poi diventa un turorial sull'assembly  :))))


.CODE

start:
                mov     ax, @data
                mov     ds, ax
                mov     es, ax
                xor     eax, 20h            ; xora il numero inserito con 20h
                cmp     Password, eax       ; è uguale alla nostra pass?
                jne     Errore              ; No? esci dal prog
                  mov     dx, offset Bravo  ; Si? Digli che è bravo
                mov     ah, 09h             ; e srivi che ha azzeccato sullo schermo
                int     21h
                mov     ah, 4ch             ; torna al DOS
                int     21h

Errore:

                mov     dx, offset Errato
                mov     ah, 09h             ; mostra la stringa "hai errato"
                  int     21h
                ret

end   start

-----------------------------------8< End code ------------------------------------
```

Bhè questo è il codice scritto alla meno peggio, comunque serve a farvi capire, se volete prendere dimistichezza con lo XOR, scaricate qui un programma che ho scritto io in C (non vi aspettate gran che) con tanto di sorgente, il programma xora un numero inserito da voi con un valore inserito da voi (il risultato lo da in decimale), divertitevi, ciauzzzzzz.

Quequero