

REPOST: SnakeOil for Real– and I am Proud of It

Source: <http://www.derkeiler.com/Newsgroups/sci.crypt/2005-10/1074.html>

From: Douglas Eagleson (eaglesondouglas_at_yahoo.com)

Date: 10/28/05

Date: 28 Oct 2005 09:23:40 -0700

*/*The Bit XOR–Bit Shifter Algorithm called SOCKET,
by Douglas Eagleson, 2005*/*

*/*I am looking for comments on the method.*

*A maximum permutation is the shift of
64factorial.*

*A bit shift to the right of the whole block and
for every shift, the shift carryover bits in the block byte set
are XOR'ed a key bit.*

*You can not get any simpler or more effect.
A larger key size can be added. And the sequential
change in key[i] as a subblock appears the perfect
algorithm. Altering block size with key size is
unnecessary. Halfway through the j loop change the
key[i] block.*/*

*/*A pad error on the last block is present*/*

*/*I have a java code MODDES that is correctly a
modern crpytographic implementation, it has the
key generation bells and pad correction details*/*

```
#include <stdio.h>  
#include <stdlib.h>
```

```
double counter;  
double limit= 1e3;;  
char ext;  
unsigned char block[8];  
unsigned char right[8];  
unsigned char left[12];  
unsigned char key[8];  
char keyin[8];  
char hello[]={ "Enter your eight char. keyboard key" };
```

sci.crypt: REPOST: SnakeOil for Real– and I am Proud of It

```
char hell[] = {"Enter your file name–follow extension rule"};
char file[12];
char file2[12];
char debud[12];
int i,j,k,s;
FILE *fele;
FILE *fele2;
int point;

void main(){

limit= 75000.0;
counter=0;

printf(hell);
gets(file);

printf(hello);
gets(keyin);

for(i=0;i<12;i++){
    file2[i]=file[i];

    if(file[i] == '.'){
        point=i;
        ext=file[i+1];
        file2[i+1]=file[i+1];
        i=12;
    }

}

if(ext=='o'){
file2[point+1]='c';

}

if(ext=='c'){
file2[point+1]='o';

}

fele2 = fopen(file2, "w");
fele = fopen(file, "r");

s=1;

while(s){

for(i=0;i<8;i++){
block[i]= getc(fele);
```

```
if(feof(fe)) {s=0;}
/*my last byte is unpadded so the lost block occurs.
An opening of the algorithm occurs because a known
carryover pads makes an alphabet. My next version will fix this
and a full implementation as a file encryptor requires
a correct key generator also.*/
```

```
}
```

```
if( ext == 'o'){
```

```
for(k=0;k<8;k++){
```

```
/*shifts the keyin[i] to right to mask bit into key[]*/
/*A string of hex as the key is the correct method
and I hope to have the key input correct soon*/
```

```
key[0] = keyin[k]&0x1;
key[1] = (keyin[k]>>1)&0x1;
key[2] = (keyin[k]>>2)&0x1;
key[3] = (keyin[k]>>3)&0x1;
key[4] = (keyin[k]>>4)&0x1;
key[5] = (keyin[k]>>5)&0x1;
key[6] = (keyin[k]>>6)&0x1;
key[7] = (keyin[k]>>7)&0x1;
```

```
counter=0;
while(counter<limit){
```

```
counter=counter+1.00;
```

```
/*save right side bit.*/
for(i=0;i<8;i++){
right[i]=block[i]&0x1;
```

```
}
```

```
/*shift block right one bit.*/
```

```
for(i=0;i<8;i++){
block[i]>>=1;
```

```
}
```

```
/*XOR key on shifted bits.*/
/*an endian implication is the symmetry of
the right set and to apply the right shift
appears dependent but in reality is independent.
```

```
A correct key application is as any bit shifted.
An xor onto any bit is possible, and I choose the
end shifted bit. Any bit would do in reality.*/
```

sci.crypt: REPOST: SnakeOil for Real– and I am Proud of It

```
for(i=0;i<8;i++){
right[i]=right[i]^key[i];

}

/*Replace rightside bits onto block.*/
if( (right[7])==0x1){
block[0]=block[0]^0x80;

}

if((right[6])==0x1){
block[7]=block[7]^0x80;

}

if((right[5])==0x1){
block[6]=block[6]^0x80;

}

if((right[4])==0x1){
block[5]=block[5]^0x80;

}

if((right[3])==0x1){
block[4]=block[4]^0x80;

}

if((right[2])==0x1){
block[3]=block[3]^0x80;

}

if((right[1])==0x1){
block[2]=block[2]^0x80;

}

if((right[0])==0x1){
block[1]=block[1]^0x80;

}
}/*end main counter loop.*/
};//end k loop

/*key Addition causes the space's appearance in relation
//to the shift. A large space.
```

sci.crypt: REPOST: SnakeOil for Real– and I am Proud of It

```
//a shift to the right of the block of eight bytes.
//and as every right bit moves a bit is XOR'ed on.*/

} //end o if

/*****
*****encrypt to decrypt*****
*****/

if( ext == 'c'){

for(k=8;k>0;k--){

/*shifts the keyin[i] to left to mask bit into key[]*/

key[0] = keyin[k-1]&0x1;
key[1] = (keyin[k-1]>>1)&0x1;
key[2] = (keyin[k-1]>>2)&0x1;
key[3] = (keyin[k-1]>>3)&0x1;
key[4] = (keyin[k-1]>>4)&0x1;
key[5] = (keyin[k-1]>>5)&0x1;
key[6] = (keyin[k-1]>>6)&0x1;
key[7] = (keyin[k-1]>>7)&0x1;

counter=0;
while(counter<limit){
counter=counter+1.00;

/*save left side bit.*/
for(i=0;i<8;i++){
left[i]=block[i]&0x80;

}

/*Change little endian of key and apply*/
if(key[7] == 0x1){
left[0]=left[0]^0x80;
}
if(key[0] == 0x1){
left[1]=left[1]^0x80;
}
if(key[1] == 0x1){
left[2]=left[2]^0x80;
}
if(key[2] == 0x1){
left[3]=left[3]^0x80;
}
if(key[3] == 0x1){
left[4]=left[4]^0x80;
}
if(key[4] == 0x1){
```

```
    left[5]=left[5]^0x80;
    }
    if(key[5] == 0x1){
    left[6]=left[6]^0x80;
    }
    if(key[6] == 0x1){
    left[7]=left[7]^0x80;
    }
/*shift block left one bit.*/
for(i=0;i<8;i++){
block[i]<<=1;

}

/*Replace leftside bits onto block.*/
if( left[0] == 0x80 ){
block[7]=block[7]^0x1;

}

if(left[1]==0x80){
block[0]=block[0]^0x1;

}

if(left[2]==0x80){
block[1]=block[1]^0x1;

}

if(left[3]==0x80){
block[2]=block[2]^0x1;

}

if(left[4]==0x80){
block[3]=block[3]^0x1;

}

if(left[5]==0x80){
block[4]=block[4]^0x1;

}

if(left[6]==0x80){
block[5]=block[5]^0x1;

}
```

sci.crypt: REPOST: SnakeOil for Real– and I am Proud of It

```
if(left[7]==0x80){
block[6]=block[6]^0x1;

}
}/*end main counter loop*/
};//end k loop

/*key Addition causes the space's appearance in relation
//to the shift. A large space.

//a shift to the right of the block of eight bytes.
//and as every right bit moves a bit is XOR'ed on.*/

};//end decrypt c if

fwrite(block,sizeof(block), 1 , fele2);

};//end big while

fclose(fele2);

};//end main

/*Note there is no array as the storage, other than
block[]. A large space is still present by use
of the XOR'ed key.*/
/*so my choice of the bits to xor make the algorithm*/

/*my hidden modified algorithm called moddes is a good
s–box removed algorithm. It does not have stupid mistakes
in it.*/
```

===== WAS CANCELLED BY =====:

Path:

...easynet–quince!easynet.net!ecngs!feeder.ecngs.de!news.germany.com!newsfeed01.sul.t–online.de!t–online.de!news

From: "Douglas Eagleson" <eaglesondouglas@yahoo.com>

Control: cancel <1130516620.692288.51890@g14g2000cwa.googlegroups.com>

Subject: Cancel "SnakeOil for Real– and I am Proud of It"

Newsgroups: sci.crypt

Date: Fri, 28 Oct 2005 17:46:55 GMT

Message–ID: <0720477173.886723.75061@g14g2000cwa.googlegroups.com>

X–Newsreader: MR/2 Internet Cruiser Edition for OS/2 v2.28a/28

X–Complaints–To: abuse@chello.at

Organization: chello.at

Lines: 2

NNTP–Posting–Host: 80.108.28.208 (80.108.28.208)

NNTP–Posting–Date: Fri, 28 Oct 2005 22:17:57 +0200

X–Trace: 42c7a43628775f5c9b97515947