Fakulteten för ekonomi, kommunikation och IT

Network Security
Laboration 0

Datum: 2007-09-03

Namn: Henrik Bäck
Marhias Andersson

Kurs: Network Security
DVGC03

Innehållsförteckning

Inledning
Ett program för att dechiffrera substitutionschiffret har skapats. Programmet presenterar alla möjliga dechiffrera för en given text.

Antaganden
För att dechiffreringen skall fungera korrekt kan den inmatade texten enbart innehålla gemener från a till z samt mellanslag. Dessa antaganden är baserade på den fakta som finns på specifikationen av denna laboration.

Detaljerad beskrivning
Programmet läser in en chiffertext från en fil. Filen lagras i ett dynamiskt allokerat minnesutrymme vilket gör att det inte finns några begränsningar i chiffrets längd.

När chiffret är inläst kan skiftning av tecken att göra. Genom att byta ut alla tecken i den inmatade filen mot ett tecken ett antal positioner bakåt i alfabetet så kan den korrekta texten erhållas. Detta upprepas så att man skiftar så många gånger som det finns tecken i det specifika alfabetet. I denna laboration blir det 25skiftningar varav eftersom en av skiftningarna, när man flyttar 0 steg, är densamma som matades in.

Alla resultat presenteras på skärmen och användaren kan gå igenom dem manuellt för att avgöra vilken som är korrekt.

Skiffer från labbspefiktion
Följande skiffer fanns att hitta i specifikationen till denna laboration.

```
max xtkebxlm dghpg nlx hy t lnulmbmnmbhg vbiaxk
tgw max lbfiexlm ptl ur cnebnl vtxltk
max vtxltk vbiaxk bgoheoxl kxietvbgz xtva exmmxk hy
max teiatuxm pbma max exmmxk lmtgwbgz makxx ietvxl
ynkmaxk whpg max teiatuxm yhk xqtfiex
ietbg fxxm fx tymxk max mhzt itkmr
vbiaxk iaap ia wbpan pda pkcw lwnpu
```

Genom att skifta denna 19 gånger erhålls den korrekta texten:

```
the earliest known use of a substitution cipher
and the simplest was by julius caesar
the caesar cipher involves replacing each letter of
the alphabet with the letter standing three places
further down the alphabet for example
plain meet me after the toga party
cipher phhw ph diwhu wkh wrjd sduwb
```

Skiffer till annan laborationsgrupp
Ett skiffer har skapats för att kunna skickas till en annan labbgrupp. Labgruppen som motog chiffret var **Anders Ellvin** och **Tobias Pulls.** De erhöll ett skiffer som hade skiftats 8 gånger.

```
lmiz abcxql
pmzm qa bw bpm kzihg wvma
bpm uqanqba bpm zmjmta bpm bzwcjtmuismza
bpm zwcvl xmoa qv bpm aycizm pwtma
```

```
bpm wvma epw amm bpqvoa lqnnmzmvbtg
bpmg izm vwb nwcvl wn zctma
ivl bpmg pidm vw zmaxmkb nwz bpm abibca ycw
gwc kiv xziqam bpmu lqaiozmm eqbp bpmu
ycwbm bpmu lqajmtqmdm bpmu otwzqng wz dqtqng bpmu
ijwcb bpm wvtg bpqvo bpqvo gwc kiv vwb lw qa bw qovwzm bpmu
jmkicam bpmg kpivom bpqvoa

bism kizm
abmdm rwja
```

Vilket om man dechiffrerar får följande text:

```
dear stupid
here is to the crazy ones
the misfits the rebels the troublemakers
the round pegs in the square holes
the ones who see things differently
they are not found of rules
and they have no respect for the status quo
you can praise them disagree with them
quote them disbelieve them glorify or vilify them
about the only thing thing you can not do is to ignore them
because they change things
```

Skiffer från annan laborationsgrupp

Det mottogs även ett chiffer från den ovannämnda gruppen som följer.

```
oczmz vmz orj otkzn ja zixmtkodji: jiz ocvo rdgg kmzqzio tjpm
ndnozm amjh mzvydib tjpm ydvmt viy jiz ocvo rdgg kmzqzio tjpm
bjqzmihzio
```

Genom att köra detta genom programmet erhölls följande text vid 21 skiftningar.

```
there are two types of encryption: one that will prevent your sis-
ter from reading your diary and one that will prevent your govern-
ment
```

Slutsats

Substitutionschiffret är mycket enkelt att knäcka. Det kräver inte många minuters tvrksamhet för att hitta den korrekta lösningen om man har en grundläggande kunskap om vad det är för språk texten egentligen är skriven på.

Funna nycklar

I laborationsspecifikationen fanns ett chiffer med nyckel **19** och i chiffret från den andra laborationsgruppen var chiffrets nyckel **21.**

Nyckeln på vårt egna chiffer, som vi gav till den andra laborationsgruppen, var **8**.

Bilaga - Programkod

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

```
#include <ctype.h>

#define ALPHALEN 26

char shiftn(const char bokstav, const int shift)
{

      return (((bokstav - 97) + shift) % ALPHALEN) + 97;

}

int main (int argc, const char * argv[])
{

      if(argc != 2)
      {

            printf ("\nDekrypteringsprogrammet\n=========\n\n");
            printf ("Användingsmönster:  %s  fileToDecrypt\n\n", argv[0]);

      }
      else
      {
            //Att läsa in till
            char * cipher;

            //Att spara alla dekrypterade varianter i
            char * decrypted[ALPHALEN];

            //Läs in fil med shiffer
            FILE * pFile;
            long lSize;
            size_t result;

            //Loopmuppar
            int i,j;

            printf ("\nDekrypteringsprogrammet\n============\n\n");

            printf ("Läser in fil: %s \n", argv[1]);

            //Öppna filen
            pFile = fopen (argv[1] , "r" );

            //Filen är finns inte
            if (pFile==NULL)
                  {
                     fputs ("File not found", stderr);
                     exit (1);
                  }

            // Hämta filens storlek
            fseek (pFile , 0 , SEEK_END);
            lSize = ftell (pFile);
            rewind (pFile);

            //Allokera minne för data
            cipher = (char*) malloc (sizeof(char)*lSize);
            if (cipher == NULL)
                  {
                    fputs ("Memory error at cipher",stderr);
                    exit (2);
                  }

            //Läs in filen till cipher
            result = fread (cipher,1,lSize,pFile);
            if (result != lSize)
```

```
                {
                  fputs ("Reading error at result",stderr);
                  exit (3);
                }

          //Dekryptera skiten
          for(i = 0; i < ALPHALEN; i = i + 1)
          {

                decrypted[i] = (char*) malloc (sizeof(char)*lSize);

                if (decrypted[i]  == NULL)
                     {
                       fputs ("Memory error at decrypted[i]",stderr);
                       exit (2);
                     }

                for(j = 0; j < lSize; j = j + 1)
                {

                     if(isalpha(cipher[j]))
                          decrypted[i][j] = shiftn(cipher[j],i);
                     else
                          decrypted[i][j] = cipher[j];
                }
          }

          for(i = ALPHALEN; i > 0; i = i - 1)
          {

                printf("----------------------\n\n(#%i)\n", ALPHALEN - i);

                printf(decrypted[i]);

                printf("\n\n");

          }


          //Radera lite
          for(i = 0; i < ALPHALEN; i = i + 1)
                free (decrypted[i]);
          free (cipher);

          //Stäng fil
          fclose (pFile);

     }

    return 0;
}
```