**DDU cybersikkerhed 6-jan-2023**

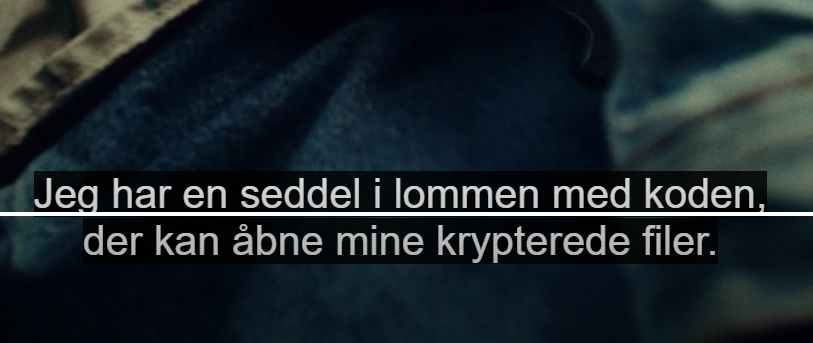
Lidt fra DR:

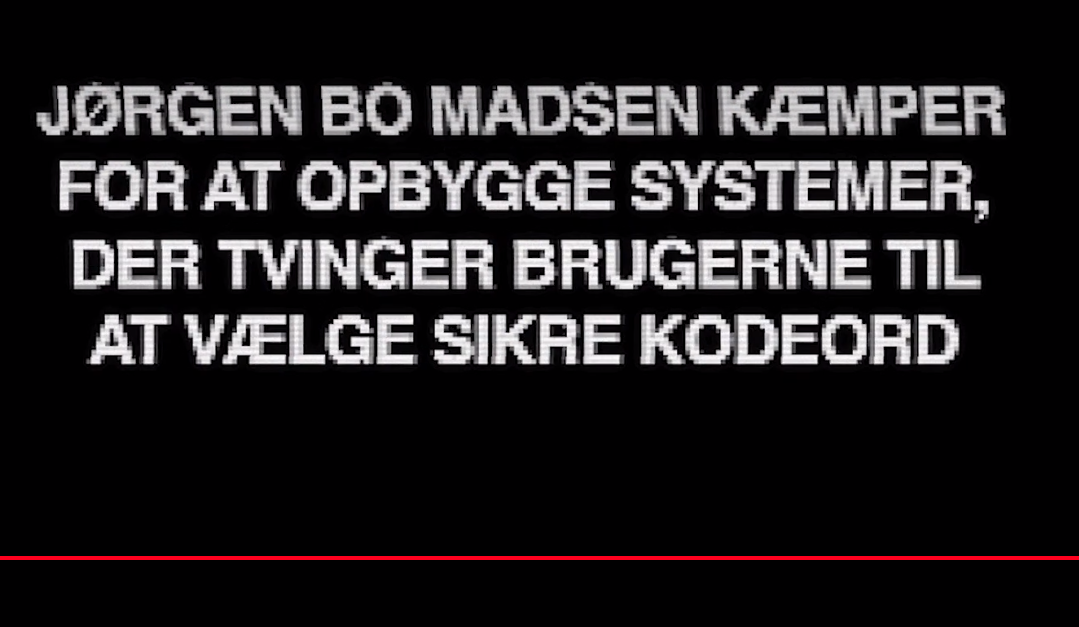
[Drengen der hackede Pentagon | Se online her | DRTV](https://www.dr.dk/drtv/serie/drengen-der-hackede-pentagon_325225) 4 gange ca 30 min

Omhandler en dansk gymnasieelevs hackning af Pentagon og blandt andet også om politimanden der efterforskede sagen, Jørgen Bo Madsen.

Ligger på DR indtil 10 april 2023

Lidt inde I 4. Afsnit:



Henne mod slutningen af sidste afsnit:  


RSA igen:

RSA beginner er rettet så den burde virke nu. Tak til Daniel og de andre der påpegede at jeg havde klokket i den. Copy og paste problem, men det har I jo nok aldrig været ude for selv.

Den er blevet oploadet på skemabrikken igen.

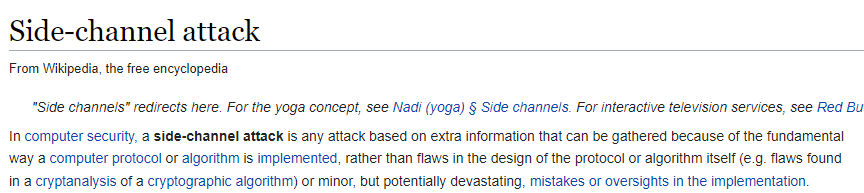
I skrivende øjeblik: 6-jan-2023 ved jeg ikke om den når at blive rettet på Haaukins inden timen her.

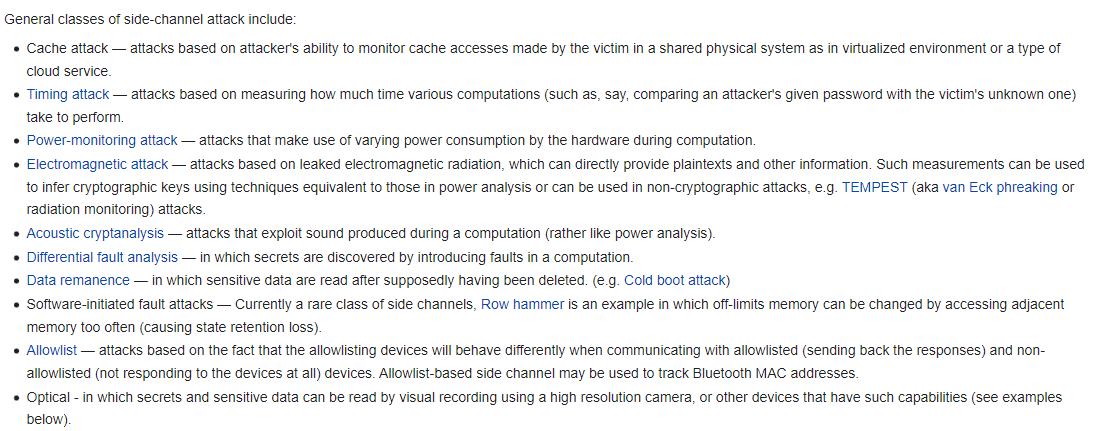
RSA brydningsforsøg:

Der har været (og er stadig) mange forskellige forsøg på at bryde RSA.

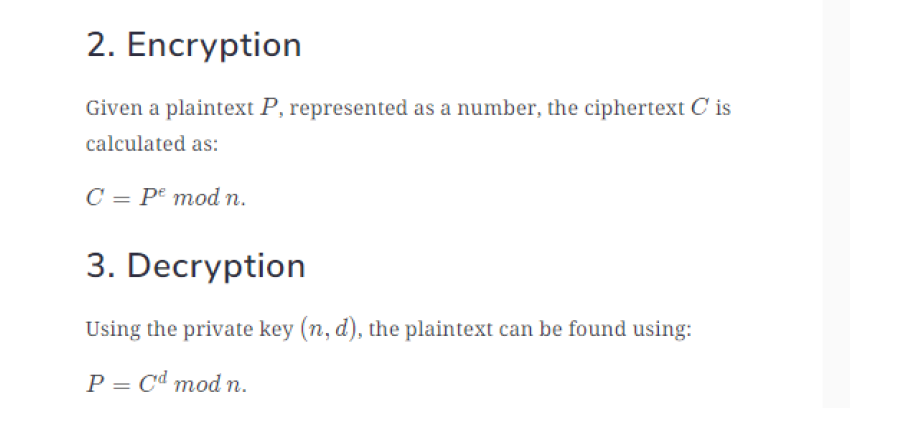
Nogle af dem har slet ikke noget med matematikken at gøre men er noget der hedder side-channel attacks.

<https://en.wikipedia.org/wiki/Side-channel_attack>





**RSA hurtig exponentiation og power analysis**



Hvis man med normale relle tal opløfter til potenser bruges logaritmeregning og så fås let et svar men det vil altid være afrundet til et antal decimaler. Det er under normale omstændigheder glimrende.

Men i kryptering regner man med heltal og udregningen skal være helt præcist ellers går det galt.  
Hvis man udregner et stor værdi på 100’vis af decimaler skal de alle være korrekte, ellers virker krypteringe/dekrypteringe ikke.

Derfor er det meget sværere at udregne store tal opløftet en kæmpestor potens.  
Hvis man bare ganger derudaf, så vil det tage alt for lang tid nærmest forevigt selv på store computere.

Til det brug er der fundet en smart måde der hedder ”square and multiply”

Det skal forstås således af man for at udregner en kæmpestor eksponentiering bruger en række af de to metoder indtil man får det rigtige resultat.

Hvis man foreksempel skal løse opgaven:

Kan gøres således:

Man starter med tallet 17 og laver en square and multiply

Så laver man en square

Så laver man endnu en square

Endelig laver man en square and multiply

Forholdsvis få operationer. Exponenten bliver kæmpestor bliver der flere operationer måske mere end 1000, men det er stadig et overkommeligt arbejde for en computerchip.

25 er i binary: 11001

Metoden til at beregne en kæmpestor eksponentiering er:

Start fra venstre mod højre i det binære og ignorer det 1 tal som det altid vil starte med.

Derefter tages resten af bittene i rækkefølge og

1 betyder square and multiply  
0 betyder square

Faktisk ret smart

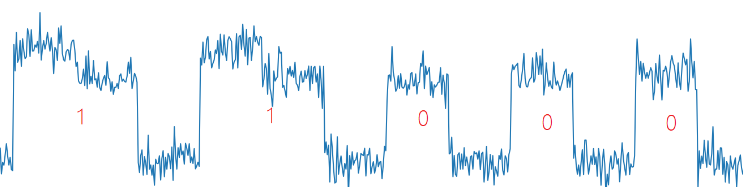
Til gengæld er det blevet brugt til at bryde RSA da strømforbruget på chips er større ved en square and multiply end den er ved en square.  
Det kan måles hvis man har fået fat i en devise som skal dekrypte RSA, for eksempel adgangskort eller lignende.

Dekrypteringen indeholder nemlig hvor C er det krypterede tal og d er den hemmelige nøgle.

En power trace kan for eksempel se således ud og den kan man lave om til et bitmønste og det nedenstående bliver så til 11000.  
Hvis man skal finde d her så sætter man et 1 tal foran (fordi det blev ignoreret i udregningen) og så fås 111000 i binary 32+16+8+0+0+0 = 56

Så i dette tilfælde er d 56 og da man kender den offentlige nøgle n og den transmiterede værdi så kan man faktisk dekryptere teksten.  
  
Vildt foruroligende.

Bemærk at hvis man skal regne med rester tages resten ved hver udregning i stedet for at vente til sidst. Det andet ville slet ikke kunne lade sig gøre ved store tal.



Denne webside er god til store tal. Jeg har i hvert fald brugt den for nyligt.

<https://www.mobilefish.com/services/big_number/big_number.php>

Opgaven RSA power analysis handler om denne metode.

**RSA den offentlige nøgle e er for lille**

Et andet attack på RSA ved uforsigtig implementation kan være at man bruger store nøgler men et meget lille e. Det er ikke principielt forbudt, men det kan i nogle tilfælde brydes meget simpelt.  
Det mindste tilladte e er 3 og det bruges faktisk i mange eksempler.

Der er en opgave i Haaukins som hedder noget i retning af RSA littlest e som omhandler dette.

Jeg vil ikke ødelægge opgaven, men kom og spørg til råds hvis I løber sur i det.

Der er en række andre RSA svagheder ved udvælgelse af nøglerne, men det kan I eventuelt selv læse om. Google ved alt.  
Der er en del RSA opgaver i forskellige konkurrencer og i disse opgaver er der en indbygget svaghed, ellers ville den slet ikke blive stillet.

**Diffie Hellman**

Dette var den første offentliggjorte metode til nøgledistribution (dem fra den engelske efterretningstjeneste var forud) og de er faktisk stjernerne indenfor dette felt.  
Uden sikker nøgledistribution ville man slet ikke komme nogle vegne med sikker kommunikation over nettet.

Metoden er faktisk forholdsvis simpel. Den baserer sig også på restregning ligesom RSA.  
Der er faktisk mange der kalder det ”clock aritmetik” da det svarer til at spørge om hvad klokken er om foreksempel 38 timer. Hvad er den?

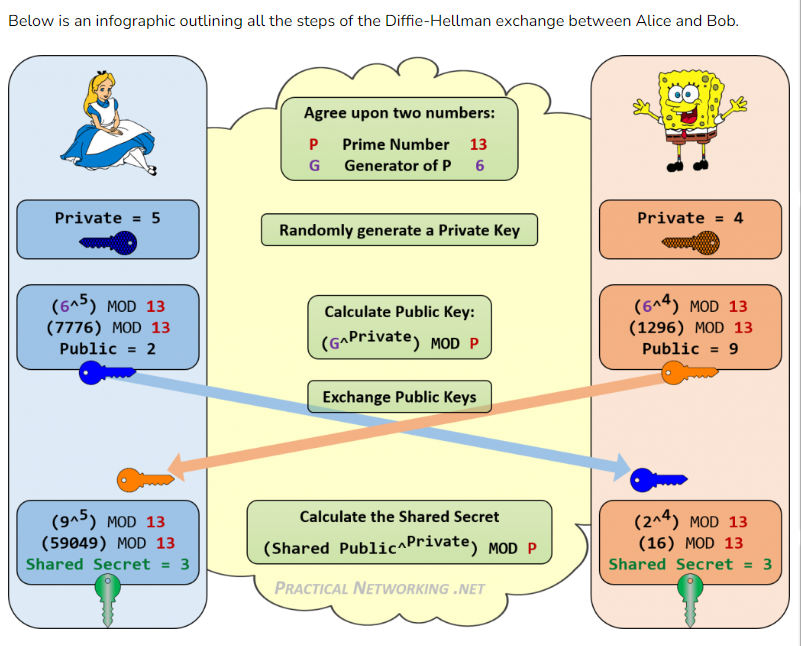
Hvordan virker det: (svarer lidt til farveblanding)  
PE viser måske på tavlen.

Alice offentlig Bob

FA FO FB Fa betyder Farve Alice osv FA og FB er hemmelige.  
 FO+FB  
 FO+FA

FO+FB+FA FO+FB+FA Disse farver er ens og kan ikke genskabes  
 let uden at kende FA

<https://www.practicalnetworking.net/series/cryptography/diffie-hellman/>



Der sendes i realiteten G\*\*A fra Alice til Bob  
og G\*\*B fra Bob til Alice  
Alice udregner så    
Bob udregner så

Ifølge potensregnereglerne giver begge samme resultat nemlig

Stort set det eneste lidt tricky er at bestemme G. Det er en såkaldt generator.  
En generator til en primtal er et tal således at man ved at tage alle tallene

osv indtil   
   
Så skal man få alle tallene fra 1 til P-1 ellers er G ikke en generator.  
Der er som regel flere G til hvert primtal

Udregn på tavlen om 6 er en generator til 13

Så har begge en hemmelig nøgle og den kan så bruges som nøgle i symmetrisk kryptering

Hvordan brydes dette.  
Hvis man ud fra den offentlige nøgle kan gætte hvad den hemmelige nøgle er så har man brudt det.  
På grund af regning med rester (modulus) kan man ikke regne baglæns men må i realiteten prøve sig frem og ved store tal tager det meget lang tid.  
Jo større P er jo længere tid tager det at bryde sikkerheden.

Der er en opgave hvor nøglen bruges til at sende et AES krypteret dokument fra Alice til Bob.  
Hvis I kan finde nøglen kan i dekryptere dokumentet.

<https://asecuritysite.com/encryption/diffie>

<https://www.dcode.fr/diffie-hellman-key-exchange>

Hvad er AES:

En ret indviklet metode til at omforme almindelig tekst til ”googiligo” som sendes til modtageren og denne kan omdanne ”googiligo” til almindelig tekst.  
I kan se noget her, men jeg er ikke klar til at forklare det i dag. Måske næste gang, hvis vi finder det interessant. Det er sådan set ikke så vigtigt hvordan det gøres. Desværre er der ikke mulighed for simpelt at lave et lille eksempel, som det var ved RSA.

<https://www.simplilearn.com/tutorials/cryptography-tutorial/aes-encryption>

AES vandt en konkurrence omkring år 2000 da man ledte efter en efterfølger for DES og tripleDES som man havde brugt indtil da.  
Konkurencen blev vundet 2 Belgiere Vincent Rijmen og John Daemen og de kaldte deres metode Rijndael . Da den vandt konkurrencen blev den officielt kaldt AES Advanced Encryption Standard

Der er mulighed for tre keylængder 128,192,256 og to modes EBC og CBC (der er faktisk 5 men jeg kender kun de 2) som er dem alle snakker om.  
Derudover kan man vælge at konvertere resultatet til enten HEX (som faktisk er BASE16)  
eller BASE64 som småt blev nævnt sidste gang.

EBC (Electronic code book) mode er den mest simple men også den mest usikre da hver blok af bits transformeres hver for sig.  
CBC mode (Ciffer Block Chaining) mode Her bruges outputtet fra den foregående blok som en slags nøgle til den næste blok, så det er sværere at lave attack på en hel tekst.

<https://www.devglan.com/online-tools/aes-encryption-decryption>

Haaukins:  
<https://hcoddu2.haaukins.com/challenges>

https://hcoddu2vpn.haaukins.com/login

Slut for i dag