2024/2025

Student: Simon Vercoutter
Klasnummer: 3
Project Titel: Breken van encryptie-algorithme
Opzichter: E. Gheysen

Breken van encryptie-algoritmes

Provinciaal Technisch instituut West-Vlaanderen

Inhoudsopgave

Hoofdstuk 1

Inleiding

1.1 Belang van deze studie

Encryptie is cruciaal in het beschermen van gevoelige informatie tegen ongewilde toegang. Van persoonlijke gegevens tot bedrijfsgeheimen en nationale veiligheid. De beveiliging van een versleuteling is slechts zo sterk als de methoden die worden gebruikt om de integriteit te waarborgen. Het testen en analyseren van encryptiesystemen is daarom van belang om zwakke plekken te onderzoeken en te verbeteren.

1.2 Uitleg over de studie

Deze studie richt zich op het onderzoeken van methoden voor het breken van encryptie, met als doel een eigen versleutelingsalgoritme te analyseren en te verbeteren.
Door de mechanismen waarmee encryptie kan worden aangevallen te begrijpen, kan worden vastgesteld of een eigen ontwikkeld algoritme bestand is tegen aanvallen.
De kern van dit onderzoek bestaat uit een combinatie van klassieke cryptanalytische technieken, zoals brute-force-aanvallen, wiskundige analyses, moderne benaderingen zoals zijaanvallen en sociale interactie.
Daarna kunnen deze methoden toegepast worden op een eigen versleutelingsalgoritme. Daarnaast wordt onderzocht hoe specifieke ontwerpkeuzes binnen het algoritme de veiligheid beïnvloeden.
De resultaten van deze studie dragen niet alleen bij aan het waarborgen van de veiligheid van het eigen versleutelingsalgoritme, maar bieden ook inzichten in de criteria waaraan moderne encryptie moet voldoen om zo goed als onkraakbaar te zijn met de kennis waarover we nu beschikken om algoritmes breken.
informatie uit (1) en (2)→zie Hoofdstuk 5

Hoofdstuk 2

Hedendaagse methodes

2.1 Brute force (brute kracht)

2.1.1 Korte uitleg

Brute force of brute kracht gebruikt de rekenkracht van de computer.
De computer gaat elke mogelijk sleutelcombinatie te proberen. Om op die manier door de versleuteling te breken.

2.1.2 Werking

De stappen die gevolgd worden zijn:

1. De computer controleert of de code mogelijk is.
2. Die combinatie wordt vervolgens ingevoerd.
3. De uitkomst wordt vergeleken met de output die ze ontvingen via het kanaal af te luisteren dat ze willen uitkomen.
4. Deze stappen worden herhaald met de volgende code.

2.1.3 Wat kan er tegen gedaan worden?

De computer doet er een langere tijd over als er meer mogelijkheden zijn.
Dus moet de encryptie sleutel zoveel mogelijk combinaties bevatten.
de gemiddelde tijd is als volgt berekend: $t_{gem} = \frac{combinaties/sec \cdot combinaties_{mogelijk}}{2}$.
Als $combinaties_{mogelijk}$ zo groot mogelijk wordt gemaakt zal t_{gem} evenredig stijgen.
informatie uit (3)→zie Hoofdstuk 5

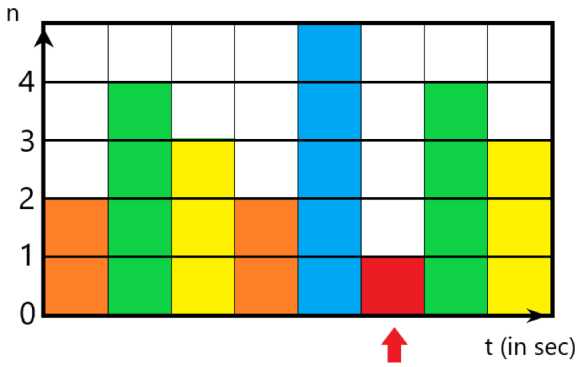
2.2 Zwakke schakels

2.2.1 Korte uitleg

In inefficiënte algoritmes zitten vaak ook zwakkere schakels.
Om een systeem te kraken moet je zoeken naar het zwakste punt, dan is er minder werk vereist.

2.2.2 Werking

Wanneer een systeem te inefficiënt is zal de computer langer doen over een stap in een vercijfering.
Omdat een computer tijd nodig heeft om dingen te berekenen zit er op momenten zwakkere encryptie op de verstuurde boodschap.
Als die tijd te lang wordt, kan een snellere computer in die tijd een sleutel hebben gevonden voor de encryptie op zijn zwakste moment.
De afbeelding is een voorbeeld van hoe sterk de beveiliging is op elk gegeven moment.
Op het rode vakje is de encryptie het zwakst waardoor de indringer daar het snelst binnen kan komen. Deze grafiek heeft een voorbeeld weer hoe sterk de encryptie is in functie van tijd:



Figuur 2.1: tabel met sterkte van enctryptie (n) in functie van tijd (1)→zie hoofdstuk 5

2.2.3 Wat kan er tegen gedaan worden?

De zwakke punten kunnen versterkt worden door een extra encryptie toe te voegen die enkel voor die stap dient om zo die encryptie extra te verstevigen. De stap erna decrypteert de computer die extra encryptie weer. Deze methode zorgt ervoor dat een interne aanval minder effect heeft door het geheugen uit te lezen.
informatie uit (4)→zie Hoofdstuk 5

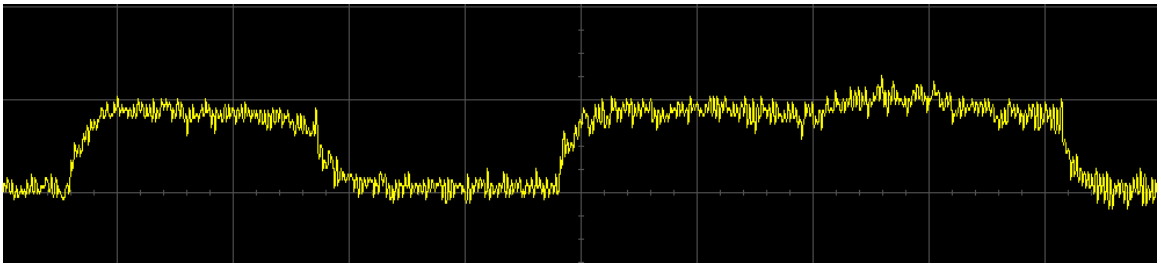
2.3 Zijaanvaltechnieken

2.3.1 Korte uitleg

Zijaanvallen zijn aanvallen die niet direct op de encryptie aanvallen maar eerder de processor om de encryptietussenstappen te achterhalen via de CPU.

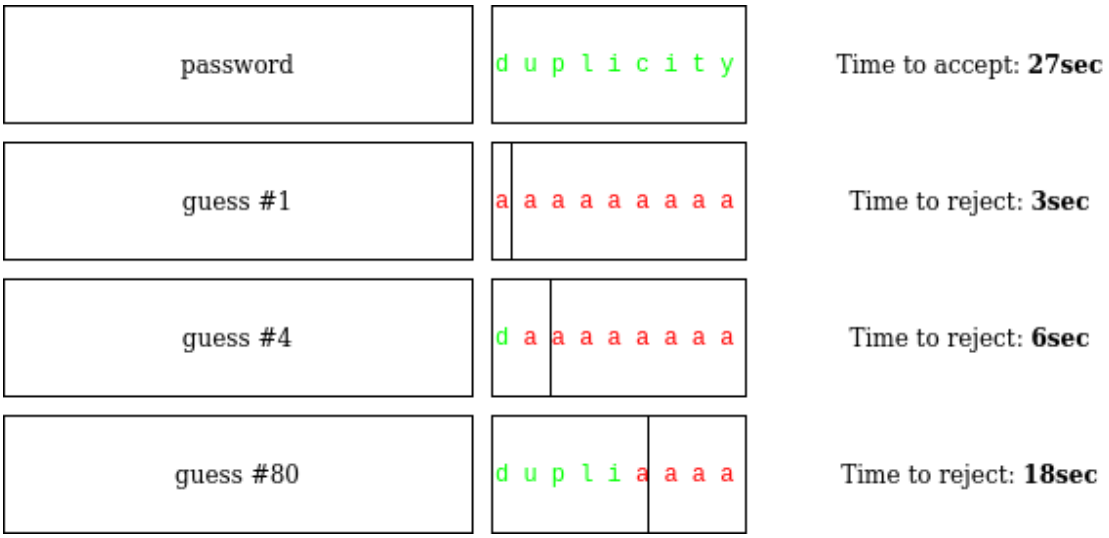
2.3.2 Werking

Het programma die de infiltratie doet vraagt frequent aan de CPU om een berekening te doen. Dit programma houdt bij hoelang de processor bezig is met deze berekening en zo weet de indringer wanneer een bit juist geplaatst is. Zo kan het programma in een bepaalde tijd de versleuteling ontsleutelen.



Figuur 2.2: grafiek van tijd/berekening in functie van tijd om activiteit in CPU te bekijken (2)→zie hoofdstuk 5

Als alles samen wordt gevoegd is dit wat visueler wanneer alle pogingen worden weergegeven zoals dit:



Figuur 2.3: tabel tijd om een paswoord te weigeren (3)→zie hoofdstuk 5

2.3.3 Wat kan er tegen gedaan worden?

Dit soort aanvallen kunnen worden geweerd door de CPU enkel bezig te houden met de encryptie en dan pas andere zaken in plaats van meerdere taken ter gelijkertijd.
informatie uit (5),(6)en(7)→zie Hoofdstuk 5

2.4 Man-in-the-middle (man tussenin)

2.4.1 Korte uitleg

Zoals de naam het uitlegt is dit een manier waarop de indringer het bericht beluistert van de verzender naar de ontvanger. Als er een zwakke versleuteling op het bericht is, kan deze persoon zelfs het bericht bewerken zonder dat de ontvanger dit weten.

2.4.2 Werking

Wanneer een bericht van de verzender vertrekt kan een derde partij dit bericht ook ontvanger door af te luisteren tussen de zendmasten. Niet enkel kan die persoon het beluisteren hij kan het bericht aanpassen zonder dat de verzender noch ontvanger dit weet als er geen authentificatie is gebruikt.
Deze manier is een van de meest voorkomende vanwege het feit dat sommige kanalen weinig tot geen encryptie hebben.

2.4.3 Wat kan er tegen gedaan worden?

Een authentificatie op het medium zou al een deel van deze problemen oplossen maar dit houdt enkel tegen dat er niets wordt veranderd. De derde partij kan in dit geval nog meeluisteren zolang er geen (sterkere) encryptie op het kanaal gebruikt wordt.
informatie uit (8)→zie Hoofdstuk 5

2.5 Wiskundige aanval

2.5.1 Korte uitleg

Dit is een nieuwere techniek om asymmetrische cryptografische algoritmen te kraken door de onderliggende wiskundige problemen op te lossen. Asymmetrische encryptie, zoals RSA, Diffie-Hellman en elliptische curve cryptografie (ECC), vertrouwt op problemen die moeilijk op te lossen zijn binnen een redelijke tijd met conventionele methoden. Mathematische aanvallen proberen deze problemen efficiënter op te lossen dan brute force.

2.5.2 Werking

Het ontsleutelingsproces maakt gebruik van een publieke sleutel (exponent en modulus) en een privésleutel die wordt berekend op basis van de priemgetallen. Deze methode gaat op zoek naar deze 2 priemgetallen met behulp van logaritmes. Want voor de sleutel ken je in dit geval: A en

C waarvoor geldt: C = A^B
dus voor B te berekenen is de volgende formule van toepassing: B = $log_a(C)$ Dit is moeilijk om in de praktijk uit te rekenen. Want een computer is niet sterk in logaritmen uitrekenen en wanneer het grondtal een priemgetal is, is deze opgave nog moeilijker.

2.5.3 Wat kan er tegen gedaan worden?

Door ongerelateerde priemgetallen of de moeilijker te kraken langere priemgetallen vermindert de kans op succesvol kraken.
informatie uit (9)→zie Hoofdstuk 5

2.6 Menselijke fouten

2.6.1 Korte uitleg

Mensen maken fouten, het is snel gebeurd.Menselijke fouten zijn verantwoordelijk voor een aanzienlijk deel van de beveiligingsinbreuken. Fouten zoals het klikken op phishing-links, het gebruik van zwakke wachtwoorden en het negeren van beveiligingsprotocollen maakt ons kwetsbaar voor cyberaanvallen.Om deze reden is dit de meest gebruikte manier om een systeem in te dringen.

2.6.2 Werking

Een mail wordt verzonden met een virus of een link naar een download die direct start om een virus te downloaden. Dit virus doordringt het systeem van de computer en stuurt alle gevonden en alle verzonden informatie door naar een derde partij, die meestal de malware heeft geïnstalleerd.

2.6.3 Wat kan er tegen gedaan worden?

Om deze risico's te verminderen, is het belangrijk om regelmatige training en bewustwordingsprogramma's voor medewerkers in te plannen. Door personeel te trainen in het herkennen van bedreigingen en het naleven van beveiligingsmaatregelen, kan de cyberbeveiliging versterken. Ook het gebruik van technische maatregelen zoals authenticatie en geavanceerde wachtwoordbeheersystemen. Deze tools kunnen helpen om de menselijke fouten te verminderen en de algemene beveiliging te verbeteren.
informatie uit (10) en (11)→zie Hoofdstuk 5

Hoofdstuk 3

Verdediging

3.1 Beginsituatie

In dit hoofdstuk wordt de situatie van de eigen ontworpen versleuteling geanalyseerd. De sleutel ziet er als volgt uit met x een willekeurig teken in een lijst van 64 tekens:

1. Er wordt een werkveld van 64 x 64 gecreëerd en in elk vak 2 letters.
2. Elke mogelijke combinatie van 2 tekens uit een lijst van 64 tekens(4096 verschillende combinaties).
3. In elke rij komt elk eerste teken van de 2 tekens, 1 keer voor.
   VB: stel ”AA” is gekozen: dan komt er geen ”Ax” meer voor in deze rij.

Een verkleinde versie ziet er als volgt uit:

	a	b	c	d	e	f	g	h	i	j
a	'cj'	'bf'	'aa'	'gc'	'ii'	'hj'	'fa'	'jh'	'ec'	'dc'
b	'bb'	'di'	'fd'	'ei'	'gi'	'if'	'cg'	'ji'	'hc'	'ab'
c	'bh'	'fe'	'gh'	'ch'	'he'	'ej'	'ac'	'jd'	'id'	'da'
d	'gf'	'hi'	'ea'	'dj'	'ca'	'ib'	'fc'	'bg'	'aj'	'jc'
e	'bi'	'eb'	'fh'	'gb'	'hg'	'dh'	'cf'	'ah'	'jg'	'ih'
f	'ad'	'ci'	'be'	'gj'	'fj'	'eg'	'dd'	'jj'	'ij'	'hf'
g	'bc'	'dg'	'ed'	'ae'	'cc'	'ha'	'jb'	'ic'	'ge'	'fi'
h	'ai'	'de'	'fb'	'cb'	'gg'	'bj'	'je'	'ie'	'hb'	'eh'
i	'cd'	'bd'	'af'	'hh'	'ff'	'ga'	'db'	'ja'	'ia'	'ef'
j	'ba'	'df'	'gd'	'fg'	'ag'	'ce'	'jf'	'ig'	'hd'	'ee'

Figuur 3.1: verkleinde sleutel (4)→zie Hoofdstuk 5

Om te versleutelen gaat de verzender eerst een willekeurige letter kiezen meestal het eerste teken van het IP-adres).
Dan zoekt hij in de rij van de eerste letter die hij wil versleutelen naar een combinatie die begint met de eerst gekozen letter.
Dan schrijft hij de 2de letter in dat vakje op. Daarna zoekt hij in de rij van de 2de letter die hij wil encrypteren naar de laatst opgeschreven letter (de 2de letter die in het laatste gezochte vakje stond. Hier schrijft de verzender weer het 2de teken op in dit vakje
En dan terug opnieuw in de rij van het derde.
VB: de verzender wil ABC versturen:
Hij kiest als eerste letter ”D” en schrijft die op. Dan zoekt hij in rij”A” naar ”DC” want ”D” is het eerste teken in het vakje. Daarna schrijft hij ”C” op. In het totaal hebben we al: DC
Voor de letter ”B” zoekt de verzender in rij”B” naar een vakje dat begint met ”C”
bijvoorbeeld: ”CG” en schrijft de laatste letter op. nu staat er al: DCG
Als laatst voor de letter ”C” zoekt de verzender in rij”C” naar een vakje dat begint met ”G” bijvoorbeeld:”GH” en schrijft de laatste letter op. Nu staat er al: DCGH en dit is het eindresultaat.

3.2 Zwakheden

In hoofdstuk 2 werd er besproken welke problemen er opduiken in verband met de methodes van in hoofdstuk 2.
De volgende problemen moeten worden opgelost:

1. De lengte van het onversleuteld bericht is altijd 1 korter dan het versleutelde bericht. Dit zorgt ervoor dat de indringer de lengte van de boodschap al kan weten.
2. Het aantal mogelijke combinatie tekens voor een teken is 64 dit is al redelijk veilig maar dit kan nog veiliger.

3. Als het IP-adres geweten is kan het eerste teken 64 keer gemakkelijker worden gevonden. Dit maakt korte berichten gemakkelijker te kraken.
4. Als 2 tekens elkaar herhalen weet de indringer dat deze tekens hetzelfde teken zijn als ervoor.
5. Omdat de volgorden van een rij niet uitmaakt is de sleutel factoriaal 64 keer gemakkelijker te kraken.
6. De sleutel opslaan kost te veel opslag. want hij moet elke teken opslaan.

3.3 Digitale versterking

Omdat er verschillende problemen waren in de oorspronkelijke code zijn er efficiënte aanpassingen gemaakt om dit probleem en mogelijks meerdere problemen op te lossen. De aanpassingen zijn:

1. Eerst werd de sleutelgenerator aangepast:
   Alle voorwaarden van in de eerste generator waren behouden maar een nieuwe regel werd toegevoegd:
   In plaats van alleen in elke rij het beginnend teken te verschillen zal de kolom dezelfde regel volgen. Hierdoor zijn al enkele problemen op gelost:
   Het aantal combinaties voor een bepaald teken zijn bijna verdubbeld, met nu 127 mogelijke combinaties.
   Als 2 tekens elkaar herhalen zijn er 2 opties waardoor de kans dat het dezelfde zijn gehalveerd is. Hierdoor is de sleutel al 2 keer moeilijker te breken.
   De volgorde van de rij maakt in deze confifuratie wel uit, omdat de kolom waarin het zich bevind een rol speelt in het versleutelen. De kans op kraken is factoriaal 64 keer kleiner geworden door deze aanpassing.
   En als laatste voordeel van deze aanpassing zal de sleutel opslag bijna halveren. Door sleutel verkorting word de eerste tekens in de eerste rij en eerste kolom genoteerd in een .txt-bestand. Dan worden alle tweede tekens van elk vakje opgeslagen waardoor de sleutel kan worden opgebouwd uit de informatie in het .txt-bestand.
2. Als tweede aanpassing werd er aan salting gedaan. Dat is het willekeurig toevoegen van tekens die later gedefinieerd worden als onbelangrijk. Dit is een snellere manier tegen zij-aanvallen dan het willekeurig berekenen van sommen om de CPU af te leiden.
   Dit verlengt de boodschap ook waardoor het versleutelde bericht niet 1 teken langer is dan het bericht.
3. En als laatste aanpassing werd de eerste letter van het versleutelde bericht een samenvoeging van alle waarden van de letters die dan weer verkort wordt naar 1 teken.
   Een extra voordeel is dat dit een controle kan zijn dat er niets is aangepast.

Hoofdstuk 4

Conclusie

4.1 Wat is er veranderd?

Om de verschillende zwakheden uit de code te halen is er heel wat kansrekenen bij komen kijken. Maar om ervoor te zorgen dat er zo weinig mogelijk kans was dat er een van bovenstaande methoden effectief zou kunnen werken is de versleuteling zo goed mogelijk afgerond.
Brute force zou gemiddeld $\frac{10^{6453}}{2}$ jaar duren. Dit is berekend van $\frac{!64^{64+2}}{2 \cdot 3{,}15 \cdot 10^{14} pogingen/jaar}$ (10 miljoen pogingen /sec).
Tegen zwakke schakels heb ik weinig kunnen doen behalve de code versnellen en een tussenencryptie versterken in een zwakkere stap.
Voor zijaanvallen zitten er willekeurige berekeningen in die weinig te maken hebben met de versleuteling. Maar dit heeft als nadeel dat de code gemiddeld 4.8% trager werkt. Dit zou de zij-aanval moeten afweren.
Een andere manier die geprobeerd werd, was ”salting” het programma voegt willekeurige tekens toe in het bericht voordat het wordt verstuurd. Er wordt op voorhand afgesproken welke tekens er geen belang hebben.
In het programma werd gekozen voor de laatste optie van deze 2 methoden om wille van er minder tijdsverlies. Man-in-the-middle aanvallen zouden ook geen probleem mogen zijn vanwege de al versleutelde berichten wanneer het verzonden wordt en als er een aanpassing wordt gedaan die merkbaar is in het ontvangen bericht. Er wordt geen authentificatie aangemaakt dus dit kan nog voor problemen kunnen zorgen. Dis is echter snel op te lossen met een hash.
Wiskundige aanvallen zijn moeilijker af te weren, maar de methode die gebruikt wordt om te versleutelen is niet asymmetrisch waardoor dit niet meer effect heeft dan brute force. Als laatste artikel van de opsomming van het tweede hoofdstuk: menselijke fouten kunnen er niet gemaakt worden omdat de mensen geen toegang hebben tot de sleutel. En extra malware past niet meer op het gebruikte programeerbord (Raspberry Pi pico 2) vanwege te weinig opslagcapaciteit.
Via deze weg zijn er geen infiltraties mogelijk.
Na alles te overwegen is de meest efficiënte oplossing gekozen in termen van snelheid en veiligheid met meer nadruk op veiligheid.

Hoofdstuk 5

Bibliografie

5.1 Informatiebronnen

1. Burge, S. (2024, 1 maart). 8 Types of Attack in Cryptography. International Security Journal. https://internationalsecurityjournal.com/types-of-attack-in-cryptography/
2. SSL: Secure Socket Layer | Ethical Hacking. (z.d.). https://www.greycampus.com/opencampus/ethical-hacking/ssl-secure-socket-layer
3. Wikipedia-bijdragers. (2024, 3 december). Brute force (methode). Wikipedia. https://nl.wikipedia.org/wiki/Brute_force_(methode)
4. Tribal, & Tribal. (2024, 15 oktober). De gevaren van zwakke schakels in de beveiligingsketen. Kennisportal Gratis kennis voor IT en Business. https://www.kennisportal.com/de-gevaren-van-zwakke-schakels-in-de-beveiligingsketen/
5. Wikipedia contributors. (2024, 24 december). Side-channel attack. Wikipedia. https://en.wikipedia.org/wiki/Side-channel_attack
6. Wright, G. (2023, 28 november). timing attack. Search Security. https://www.techtarget.com/searchsecurity/definition/timing-attack#::text=A%20timing%20attack%20is%20a,attacks%20against%20some%20encryption%20methods.
7. Check Point Research. (2024, 7 juli). Modern Cryptographic Attacks: A Guide for the Perplexed. https://research.checkpoint.com/2024/modern-cryptographic-attacks-a-guide-for-the-perplexed/
8. Wikipedia-bijdragers. (2024, september 15). Man-in-the-middle-aanval. Wikipedia. https://nl.wikipedia.org/wiki/Man-in-the-middle-aanval
9. https://www.cs.purdue.edu/homes/ninghui/courses/Fall05/lectures/355_Fall05_lect23.pdf, (2024, January 21)
10. Ahola, M. (2021, 14 september). De rol van menselijke fout bij succesvolle inbreuken op de cyberbeveiliging. De Rol van Menselijke Fout bij Succesvolle Inbreuken op de Cyberbeveiliging. Geraadpleegd op 14 januari 2025, van https://blog.usecure.io/nl/de-rol-van-menselijke-fout-bij-succesvolle-inbreuken-op-de-cyberbeveiliging
11. Guez, Y. (z.d.). Crypteron. https://www.crypteron.com/blog/the-real-problem-with-encryption/

5.2 Afbeeldingen

1. Eigen tabel met een imaginair voorbeeld van hoe sterk een encryptie is in functie van tijd.
2. Wikipedia contributors. (2024, december 24). Side-channel attack. Wikipedia. https://en.wikipedia.org/wiki/Side-channel_attack/media/File:Power_attack.png
3. Check Point Research. (2024, 7 juli). Modern Cryptographic Attacks: A Guide for the Perplexed. https://research.checkpoint.com/2024/modern-cryptographic-attacks-a-guide-for-the-perplexed/ Figure 11 – Progression of a timing attack.
4. gegenereerde sleutel met een generator, Simon Vercoutter, (2024, January 20)