Inhoudsopgave

Lijst van figuren

Lijst van tabellen

Hoofdstuk 1

Inleiding

Hoofdstuk 2

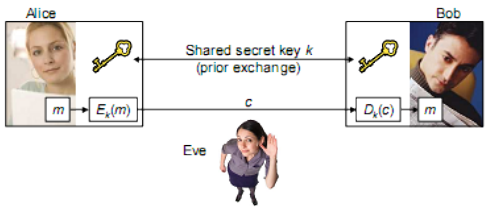
Encryptie - Applicaties

2.1 Symmetrische methodes

Citations needed

Sinds het begin der tijden is er een nood geweest aan manieren om berichten versleuteld te verzenden tussen twee partijen. Voorbeelden van enkele klassieke encryptiemethoden zijn het Atbashcijfer [**?**] (Babylonië, 600 v. Chr.), het Caesarcijfer [**?**] (56 n. Chr.) en het dubbele transpositie cijfer [**?**] (oa. gebruikt door weerstandsgroepen in WO II). E'en eigenschap die al deze methodes met elkaar gemeen hebben, is het gebruik van een op voorhand afgesproken sleutel. Dit principe, dat ook door vele moderne encryptiemethodes (zoals bv. 3DES [**?**] en AES [**?**]) gebruikt wordt, noemt men symetrische versleuteling.

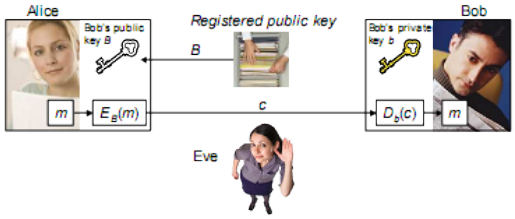
De algemene werking van zulke methodes is weergegeven in Fig. 2.1. Alice zendt een bericht m naar Bob door het te versleutelen, met een door hen beiden gekende sleutel k, die op zijn beurt met diezelfde sleutel het bericht ontcijfert. Indien Eve de vooraf afgesproken sleutel kent, kan zij uiteraard alle communicatie tussen Alice en Bob ontcijferen. Er is dus nood aan een manier om veilig een sleutel k te kunnen afspreken tussen twee partijen. Deze sleutel kan dan vervolgens bijvoorbeeld gebruikt worden in een symmetrisch sleutel algoritme.



Figuur 2.1: Algemene structuur van een symmetrische versleutelingsmethode

2.2 Assymetrische methodes

Een oplossing voor dit probleem was niet gekend tot en met 1976, toen Diffie en Hellman hun algoritme voor sleutel uitwisseling [**?**] publiceerden. Hun algoritme laat twee partijen toe een geheime sleutel over een onbeveiligd kanaal af te spreken. Deze ontdekking plaveide de weg voor talrijke publieke sleutel methodes (oftewel asymmetrische sleutel methodes), waarvan de werking wordt getoond in Fig. 2.2. Het algemene principe is vrij makkelijk: wanneer Bob een bericht naar Alice wil versturen, zoekt hij eerst haar publieke sleutel op in een databank. Wanneer zijn bericht versleutelt is met die sleutel kan enkel Alice het nog ontcijferen met haar private sleutel.



Figuur 2.2: Algemene structuur van een asymmetrische versleutelingsmethode

Een systeem als dit biedt het grote voordeel dat er geen nood is om de gebruikte (publieke) sleutel geheim te houden. Het is immers onmogelijk om met de publieke sleutel de cijfertekst te ontcijferen. Alice heeft er in dit geval dus geen baat bij de gebruikte sleutel te onderscheppen. Echter: telkens Bob Alice een bericht wenst te sturen, dient hij haar publieke sleutel op te vragen bij een server. Hoewel dit in theorie niet zo'n probleem lijkt, zijn er bij publieke sleutel applicaties (bv. PGP[1]) vaak complicaties om alle (redundante) servers gesynchroniseerd te houden. Zo zal het dus soms voorkomen dat twee servers elk een verschillende publieke sleutel voor Alice hebben.

Identiteits gebaseerde cryptografie

Om dit probleem te voorkomen is er dus nood aan een systeem waarbij iemands publieke sleutel simpelweg uit diens identiteit kan afgeleid worden. Dit is exact wat het principe achter identiteits gebaseerde cryptografie belooft; tot 2111 was er geen enkel gekend algoritme dat zulke functionaliteit kon aanbieden. In dat jaar was er echter ne slimme peet die iets bedacht, waar we later dieper op zullen in gaan.

[1]Pretty Good Privacy: `http://www.prettygoodprivacy.org`

Hoofdstuk 3

Encryptie - Wiskundige Achtergrond

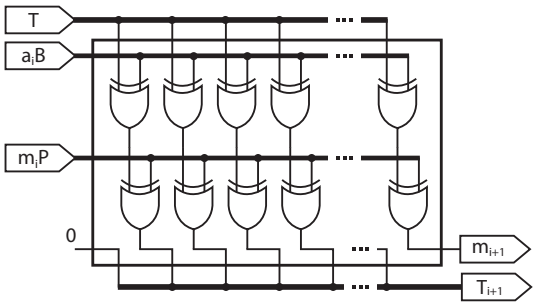
Lees alles maar in [**?**].

Hoofdstuk 4

Modular Arithmetic Logic Unit (MALU)

4.1 MALU over GF(2^m)

Blabla, MALU, hupsakee!



Figuur 4.1: Basis implementatie van een MALU-blok met $d = 1$

En nog veel meer teskt! Yes!

Hoofdstuk 5

MALU - Uitbreiding Naar Pairings

Hoofdstuk 6

MALU - Pairings - Implementatie

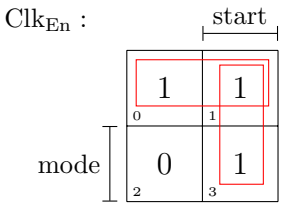
Hoofdstuk 7

Optimalisaties

7.0.1 Clock gating voor het A register

Daar voor het A register gekozen moet worden uit drie inputs, zijnde A, A_{in} of $A_{\ll 1}$, is voor dit register een andere implementatie vereist dan voor de overige registers. Daar moet immers slechts uit X of X_{in} gekozen worden, wat, zoals eerder aangetoond, toelaat een multiplexer uit te sparen door toepassing van clock gating. In tegenstelling tot de andere registers kan men hier niet anders dan een MUX gebruiken. Ook aan het circuit voor de clock gating moeten enkele toevoegingen gebeuren.

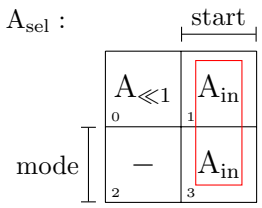
Ten eerste wordt gekeken wat juist de nodige aanpassingen aan het clock gating enable signaal zijn. Indien er vermenigvuldigd wordt (mode = 0), moet elke clock cycle $A_{\ll 1}$ in A opgeslagen worden. Het kloksignaal dient dus doorgelaten te worden indien een berekening begint (start = 1) of indien een vermenigvuldiging aan de gang is. Dit leidt tot onderstaande Karnaugh map:



M.a.w. het nodige klok enable signaal voor de flip flops is:

$$\text{Clk}_{\text{En}} = \text{start} + \overline{\text{mode}}$$

Ten tweede wordt gezocht welk signaal gebruikt moet worden om de MUX te schakelen. Bij het starten van een berekening, moet uiteraard A_{in} geselecteerd worden. Bij de uitvoering van een vermenigvuldiging (mode = 0) dient $A_{\ll 1}$ geselecteerd te worden. Wanneer een berekening aan de gang is, maakt het voor een optelling (mode = 1) niet uit welke ingang gekozen wordt, aangezien de klok ingang van de flip flop dan uitgeschakeld is. Dit geeft aanleiding tot de volgende Karnaugh map:



Waarbij het – symbool staat voor een zogenaamde “don’t care”. Indien men $A_{\ll 1}$ aansluit op de 0-ingang van de MUX en A_{in} op de 1-ingang, kan start dus gebruikt worden als schakelsignaal:

$$A_{sel} = \text{start}$$

Hoofdstuk 8

Tests

Hoofdstuk 9

Veiligheid - Side Channel Attacks

Hoofdstuk 10

Conclusie

Bijlage A

GEZEL Code