Effektanalysattacker
Del II: Döljning

Anders

19 juli 2022 (23:51)

Sammanfattning

Häri beskrivs döljning som försvar mot effektanalysattacker såväl som attacker mot döljning.

Innehåll

0 Inledning

1 Avgränsningar

Döljning kan referera till alla möjliga sidokanaler, men i denna artikel begränsar vi oss till effektanalysattacker.

2 Syfte

Syftet är att inte låta en anfallare erhålla information som kan vara till fördel vid gissandet av hemliga data.

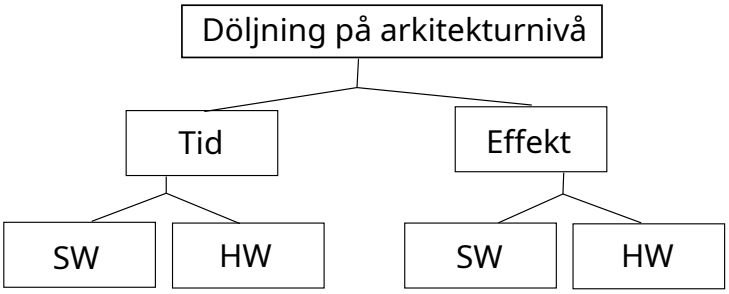
Vad man vill åstadkomma med döljning är att konstruera en apparat som inte röjer information om data som behandlas. Alltså vill man att effektförbrukningen inte ska bero på vilka operationer som utförs eller vad mellanstegsvärdena är under exekveringen av en algoritm, typiskt en kryptografisk algoritm.

3 Former av döljning

Rent naivt kan man tänka sig några sätt att dölja på. Vid en effektmätning tar man hänsyn till två dimensioner; varje mätvärde består av en tidpunkt och en effekt (kanske går man omvägen genom en ström eller tidsderivatan av ett magnetiskt flöde eller liknande). Det är alltså dessa två dimensioner vi kan påverka. Man skulle kunna konstruera apparaten sådan att …

- …dess effektförbrukning är konstant från klockcykel till klockcykel
- …dess effektförbrukning är slumpmässig från klockcykel till klockcykel
- …den förskjuter operationer slumpmässigt i tiden
- …den utför *låtsasoperationer* (*eng*: dummy operations), vars resultat inte påverkar algoritmen
- …den kastar om ordningen på operationerna
- …dess effektförbrukning är svårmätt genom att minska SNR
  - öka bruset genom att slumpmässiga omställningar dominerar effektförbrukningen
  - dämpa signalers effektförbrukning

Den första delen av denna artikel kommer ägnas åt att studera hur dessa metoder kan implementeras. Sedan kommer vi diskutera hur de kan anfallas.



Figur 0: Överblick för indelning av döljning på arkitekturnivå.

Låt oss börja med att betrakta döljning på arkitekturnivå och sedan på cellnivå.

4 Döljning på arkitekturnivå

Arkitekturnivån är den högre av de två nivåerna som behandlas. På arkitekturnivån studerar vi hur en given algoritm implementeras i digital elektronik, antingen som sekvenser av instruktioner i programkod som körs på en mikroprocessor eller som konfiguration av logikelement och kopplingarna mellan dem.

Vi delar grovt upp döljning i de två dimensionerna, tid och effekt, och sedan finare i mjukvaruimplementationer (SW) och hårdvaruimplementationer (HW)[†]. Låt oss börja med döljningsmetoder i tidsdimensionen.

4.0 Döljning på arkitekturnivå i tidsdimensionen

Låt oss här, ännu finare, dela in döljningsförfarandena i två kategorier, mjukvara och hårdvara.

4.0.0 Döljning på arkitekturnivå i tidsdimensionen för mjukvara

För mikroprocessorer är det inte så lätt; det typiska är att man försöker dölja sina förehavanden genom att skjuta in låtsasoperationer och kasta om ordningen i vilken operationer utförs. Dessa metoder kräver, för att de inte ska vara förutsägbara för en anfallare, vara slumpartade. För att uppnå det krävs slumptal som typiskt genereras inuti apparaten, se avsnitt D.

[†]Mjukvara syftar här på att algoritmen exekveras på en mer allmän mikroprocessor, som har förmågan att exekvera godtycklig programkod som definieras av en serie av instruktioner. Hårdvara syftar här på att algoritmen exekveras på mer specialiserad elektronik, typiskt ASIC- eller FPGA-teknologier. Dessa kan betraktas som kretsar av logik, i termer av digitala elektronikkomponenter, som definierar algoritmen. I det senare fallet består den av logik som kan programmeras om.

4.0.1 Döljning på arkitekturnivå i tidsdimensionen för hårdvara

För hårdvara har vi fler möjligheter. Utöver de metorder vi hade för mjukvara så har vi även möjlighet att trixa med klockan.

Som vi nämnt så fungerar låtsasoperationer och omkastning av ordning av operationer på exakt samma sätt. Vi kan för hårdvara även skjuta in en låtsascykel (*eng*: dummy cycle), som kan ske mitt under en operation. Denna cykel kommer alltså förlänga operationen med en klockcykel.

En utveckling av låtsascykler är att ha dubbla register för varje mellanstegsvärde och exekvera samma algoritm för en uppsättning av äkta värden och en uppsättning låtsasvärden, vilket man ekvivalent kan betrakta som att man exekverar en äkta algoritm och en identisk, så när som på operanderna, låtsasalgoritm. Man kan då låta slump avgöra huruvida en klockcykel avancerar algoritmen och sparar värdena för den äkta algoritmen eller för låtsasalgoritmen.

Ett mycket simpelt klocktrix kan vara att skippa klockpulser ibland. Låt ett slumptal som genereras varje klockcykel bestämma om nästa klockcykel ska skippas, genom att dra klocksignalen låg under hela cykeln, eller ej.

Man kan även slumpvis förändra klockfrekvensen under körning. Detta kan såklart kräva att man har marginal i kretsens timing[†]. En annan metod vore att låta apparaten slumpvis skifta mellan olika frekvenser för olika klockdomäner på samma krets. Detta kräver en del förtanke i designen.

För att dessa döljningsförfaranden ska fungera krävs det att en anfallare inte kan lära sig att identifiera dem. Om en anfallare kan känna igen en låtsascykel eller en klockfrekvensförändring så är de inte längre lika hjälpsamma för att dölja skyddsvärda data.

Precis som för mjukvarufallet kräver dessa metoder god slump, se avsnitt D.

4.1 Döljning på arkitekturnivå i effektdimensionen

Precis som förut delar vi in i mjukvara och hårdvara.

4.1.0 Döljning på arkitekturnivå i effektdimensionen för mjukvara

Givet en algoritm är det främsta sättet att påverka effektförbrukningen i mjukvara valet av instruktioner som utgör programkoden som denna algoritm beskrivs av. En funktion kan implementers i två olika sekvenser av instruktioner

[†]Det är lätt att tänka sig att saker sker omedelbart, men det är inte så i praktiken. Det tar tid för signaler i en krets att ta sig igenom logiska element, så som grindar och buffrar. Vidare har ledarna längd, vilket lägger på ytterligare tid för att signaler ska propagera. Vid design av en digital krets måste dessa saker tas hänsyn till, så att signalerna mellan varje klockcykel har hunnit stabilisera sig till de värden som definieras av designen. Detta kallas för timing, och att ha marginal på en tid, Δt, innebär att signalerna har varit stabila i Δt innan nästa cykel börjar.

sådana att de är ekvivalenta, men läcker olika mycket information om resultatet och operanderna i termer av effektförbrukning.

Särskilda instruktionssekvenser i ett program som beror av nyckeln (exempelvis en konditionell jump-instruktion som beror av nyckeln) bör undvika eftersom en anfallare kan lära sig att känna igen effektförbrukningsmönstret. På samma sätt, om minnesadresser beror av nyckeln bör de väljas så att de läcker så litet information som möjligt (konstant hammingvikt eller konstant hammingavstånd).

Det är en möjlghet ann sänka SNR genom att parallellt till algoritmen som exekveras också beräkna annat (som man inte nödvändigtvis bryr sig om resultatet av). För detta kan annat än bara mikroprocessorn som algoritmen exekveras på användas, utan även andra delar så som koprocessorer och gränssnitt.

4.1.1 Döljning på arkitekturnivå i effektdimensionen för hårdvara

Precis som i tidsdimensionsfallet har vi här mer friheter än för mjukvara. Det är exempelvis rimligt att göra effektförbrukningen lika från klockcykel till klockcykel snarare än bara slumpartad. Alla metoder som gällde för mjukvara gäller också för hårdvara.

Utöver dessa kan vi även använda filter som vi sätter mellan spänningsmatningen och den kretsen som exekverar algoritmerna. Genom att använda brytarkondensatorer och konstant-ström-källor och -dioder kan vi reglera effektförbrukningen, och, om vi vill, göra den så gott som konstant.

Vi kan även, som ett sätt att sänka SNR, införa bruskällor. Vanligen bygger man sådana av slumpgeneratorer (se avsnitt D) som man kopplar till kondensatorer. Den slumpvisa uppladdningen och urladdningen av dessa kondensatorer får stora utslag på effektförbrukningen och minskar således SNR.

Vi minns här förhållandet mellan energin, E, i en kondensator med kapacitans, C, och spänning, V:

$$
\begin{aligned}
E &= \frac{1}{2}V^2C \\
&= \frac{Q^2}{2C} \\
&= \frac{1}{2}QV \\
&= \frac{1}{2}V\int_{\Delta t} J(t)dt
\end{aligned}
\tag{0}
$$

Där Q är laddningen som den spänningen motsvarar och $J(t)$ är strömmen som laddar upp konensatorn under tidsperioden Δt.

En viktig detalj att ta hänsyn till i dessa sammanhang är att effekten inte alltid mäts vid nätaggregatet utan kan mätas genom EM-probning[†]. Därför behöver bruskällor vara utspridda över hela kretsen snarare än att sitta i ett hörn, medan algoritmen exekveras i motstående hörn.

5 Döljning på cellnivån

På cellnivå studerar vi hur logiken implementeras i det som i algoritmnivåbeskrivningen betraktades som logiska ellement. Vår önskan är att konstruera dem sådana att deras effektförbrukning är konstant mellan klockcykler, oavsett vad som händer. I andra ord kan man uttrycka det som att man vill bygga logikelement sådana att logiken förbrukar en konstant effekt, oavsett vad för operationer som utförs och på vilka operander. På så sätt kommer hela apparatens effektförbrukning vara densamma varje klockcykel.

5.0 Dubbelspårig förladdadningslogik

En metod för att uppnå konstant effektförbrukning mellan klockcykler är att bygga logikelementen sådana att de vid varje klockcykel sätter lika många ledare till hög potential som låg potential. Detta kan uppnås genom att kombinera två metoder: *förladdning* och *komplementära* signaler.

Förladdning innebär att vi väljer ett förladdningsvärde (0 eller 1). Sedan delar vi in varje klockcykel i två steg, en för förladdning och en för *utvärdering*. I vanliga fall är hela klockcykeln ett utvärderingssteg. Under förladdnigssteget sätts varje ledare till förladdningsvärdet och vid utvärderingssteget sätts ledarna till de värden som definieras av logikdesignen.

Dubbelspåriga signaler innebär att det för varje signal i kretsen går två ledare. Dessa kan användas komplementärt, alltså för varje par av ledare är en satt till den potential som definieras av logikdesignen och den andra ledaren är satt till komplementet. Detta kallas även för *differentiella* ledare. En dubbelspårig logikcell har alltså dubbelt så många ingångar och dubbelt så många utgångar som en *enkelspårig* logikcell.

Kombinerar vi dessa två metoder får vi något som kallas *dubbelspårig förladdningslogik* (*eng*: Dual Rail Precharge Logic, DRPL)[♯]. För DRPL har vi ett förladdningssteg under vilket alla ledare sätts till samma potential, och varje par av dubbelspåriga ledare är alltså lika. Sedan har vi utvärderingssteget, under vilket varje par av dubbelspåriga ledare sätts till det värde som definieras av logikdesignen respektive dess komplement.

[†]Genom att mäta, till exempel, förändringen av det magnetiska flödet igenom en sluten yta (minns här att B-fältet är konservativt, se [Fur22a] för genomgång av elektrodynamikens principer) kan man använda Lenz lag (tidsförändringen av magnetiskt flöde genom en yta som begränsas av en slinga är proportionellt mot spänningen över slingan, $V_{\text{inducerad}} = -\frac{d\Phi_B}{dt} = -\frac{d}{dt}\int dA\langle n, B\rangle$) för att avgöra variationer i ström inom ett område, säg en del av en elektronisk krets. Se [Fur22b] för diskussion om mätuppställningar för effektanalysattacker.

[♯]Ibland används förkortningen DPL där D står för Dual Rail.

Betraktar vi ett logiskt DRPL-element under en klockcykel inser vi att lika många ledare sätts höga ($0 \to 1$) som låga ($1 \to 0$) under varje klockcykel. Ett exempel. Låt oss säga att förladdningsvärdet är 1 och betrakta paret av utgångar från en logisk grind. Från mitten av förladdningssteget till mitten av förladdningssteget kommer den ena signalen att göra övergångarna $1 \to 0$ och $0 \to 1$, medan den andra kommer vara konstant 1 ($1 \to 1$ och $1 \to 1$). Detta är sant oavsett resultatet av logikgrinden, och därför förbrukas lika mycket effekt varje klockcykel oavsett operanderna och resultatet.

En naturlig fråga som uppstår är hur vi kan bygga *låskretsar* (*eng*: latches) och *vippor* (*eng*: flip-flops) med denna logikstil sådana att deras effektförbrukning är konstant från klockcykel till klockcykel. Låt oss fundera på fallet med vippor.

En möjlig lösning för att bygga en DRPL-vippa av d-typ[†] är att för varje signal som ska in i vippan byta ut vippan mot två seriekopplade vippor sådana att de triggas två gånger per cykel. På så sätt får man rätt värden vid utgångara under utvärderingssteget och likaså rätt värden under förladdningssteget. Därtill har vi alltjämt att alla ledare förladdas under förladdnignsfasen.

När vi hittills betraktat effektförbrukningen som konstant mellan klockcykler så länge lika många ledare laddas till en viss potential varje klockcykel så har det grundat sig på ett vanskligt antagande, nämligen att varje ledare har samma kapacitans. Om vi inte längre antar det så faller argumentet omkull. Vi bör därför naturligtvis balansera varje par av komplementära ledare så att de får samma kapacitans.

Det största bidraget till kapacitansen kommer från ledaren själv snarare än logikelementen den är kopplad till, så det är huvudsakligen ledarna som justeras.

Vi bör även notera att om en anfallare har mycket god upplösning i tid måste vi vara försiktiga så att ledarna laddas lika snabbt också, inte bara lika mycket. Detta innebär att vi även behöver justera ledarnas konduktans. Vi minns Ohms lag, strömtätheten, j, är proportionell mot det elektriska fältet, E, med en koefficient, σ, som kallas materialets *konduktivitet*:

$$j = \sigma E \tag{1}$$

För elektriska kretsar kan vi betrakta strömmen, $J_{a \to b}$, i en ledare mellan två noder, a och b, som proportionell mot spänningsskillnaden, $V_{a \to b}$ och en koefficient, $\Sigma_{a \to b}$, som kallas för *konduktans*:

[†]D-flip-flop, vippa av d-typ, är en typ av vippa som när den triggas, exempelvis av en stigande flank av en klocksignal, sparar ingångssignalens potential och sätter sin utgångssignal till denna potential. Utgångssignalen förblir konstant tills nästa gång vippan triggas. Detta är en mycket vanlig och användbar typ av vippa.

$$
\begin{aligned}
J_{a\to b} &= \Sigma_{a\to b} V_{a\to b} \\
&\iff \\
J_{a\to b} R_{a\to b} &= V_{a\to b}
\end{aligned}
\tag{2}
$$

Där vi infört symbolen $R_{a\to b} = 1/\Sigma_{a\to b}$ som kallas för *reistansen* mellan noderna[†]. Om detta känns obekant är det förmodligen dags att påminna sig genom att läsa [Fur22a], [Jac62] eller [Nor].

Vi kan alltså se att konduktansen påverkar hur snabbt kondensatorer laddas upp och två olika kondensatorer med samma kapacitans men olika konduktans kommer alltså förbruka lika mycket effekt under en klockcykel men de kommer inte ha samma momentana effektförbrukning under cykeln.

6 Utnyttjbara svagheter i döljning

Allmänt kan apparater som döljning är implementerat på ändå anfallas på grund av exempelvis att dålig slump används för att skjuta in låtsasoperationer eller kasta om dem, eller att de komplementära ledarna i DRPL inte är balanserade med avseende på kapacitans. Vi ska nu undersöka hur dessa svagheter kan utnyttjas.

6.0 Omkastade spår

I tidsdimensionen gick våra döljningsmetoder ut på att förskjuta operationer i tid, kasta om dem och skjuta in låtsasoperationer.

Dessa döljningsmetoder gör att spåren är omkastade. Det är mycket svårt att utnyttja sådana spår, och man tvingas mäta många gånger för att utnyttja kovariansen mellan äkta spår och omkastade spår, vilket grundar sig på slump som inte är perfekt, eller antaganden om hur döljningen påverkar spårens omkastning.

6.1 Förskjutna spår

Dessa döljningsförfaranden resulterar i förskjutna spår, alltså att ett mätvärde i ett spår inte nödvändigtvis motsvarar mätvärdet i ett annat spår även om det har motsvarande tidskoordinat. Vi bör även notera att detta inte alltid enbart är en effekt av döljning i tidsdimensionen, utan kan i praktiken bero på triggersignalen till oscilloskopet som används för att mäta. Därför är förskjutna spår ett problem även i implementationer utan döljning.

En anfallare vill ordna spåren så att han kan använda de olika spåren för att bygga korrelationskoefficienterna som her honom en fördel i gissandet av, exempelvis, den kryptografiska nyckeln som används i en kryptografisk algorimt som en apparat utför.

[†]Ibland kallas kretsversionen i termer av resistans för Ohms lag.

6.1.0 Invers omkastning av förskjutna spår

Att sortera ett omkastat spår så att det motsvarar ett oomkastat spår är inte helt lätt. Vanligtvis sker det i två steg.

0. Finn en sekvens som ger upphov till ett mönster i effektförbrukningen som går att känna igen i varje spår
1. Förskjut spåren så att mönstret får samma tidskoordinat i varje spår

6.1.0.0 Välja mönster

Att välja ett mönster är inte helt enkelt, men två huvudsakliga principer gäller:

0. Det bör vara unikt, i den bemärkelsen att mönstret inte uppkommer flera gånger per spår.
1. Det bör inte bero på data som kan vara olika per exekvering.

Om anfaller inte har stor kunskap om eller algoritmen, eller om implementationen är sådan att effektförbrukningen är konstant mellan klockcykler så kan det vara mycket svårt att finna ett användbart mönster.

6.1.0.1 Välja motförskjutning

Då anfallen har funnit ett användbart mönster behöver han förskjuta varje spår i tidsdimensionen så att spåren passar varandra. Detta görs vanligtvis med statistiska metoder i stil med:

0. *Minsta kvadratmetoden*[†].
1. Maximera korrelationskoefficienterna.

6.1.0.2 Utnyttjande av genererad slump

Om en anfallare kan hitta mönster i vilken effektförbrukningen beror av den genererade slumpen kan han välja ut spår med hänsyn till detta, och på så sätt erhålla en delmängd av spåren som härrör från en skev fördelning av slumptal, även om den genererade slumpen var likformigt fördelad.

[†]En statistisk metod för att hitta en lösning som minimerar felet för ett överbestämt linjärt ekvationssystem. Beskrivs ekvationen som $Ax = b$, där $A \in (\mathbb{R}^{\delta} \otimes (\mathbb{R}^{*})^{d}$ beskriver koefficienterna för några variabler, $x \in R^{d}$, jämfört med några värden, $b \in \mathbb{R}^{\delta}$, och $d > \delta$. Då kommer felet minimeras av ekvationen $A^{\dagger}Ax = A^{\dagger}b$, där $A^{\dagger}$ är adjunkten av A med avseende på den inre produkten, $\langle x^{\mu}\partial_{\mu}, y^{\nu}\partial_{\nu}\rangle = x^{\mu}y^{\mu}$, som induceras av den euklidiska metriken, $g_{\mu\nu}dx^{\mu} \otimes dx^{\nu} = dx^{\mu} \otimes dx^{\mu}$. Se exempelvis [Fur22c], [Spi70] eller [Sch50] för mer detaljerad beskrivning.

6.1.1 Anfalla förskjutna spår som de är

Om anfallaren inte kan finna inversa förskjutningar för sina spår så är det kanske inte helt kört ändå. Det finns några tekniker som kan användas för att motverka förskjutna spår.

Anfallaren kan dela in spåret i tidsintervall, integrera effektförbrukningen över dem och använda det resultatet. I allmänhet är det inte lätt att välja storlek på tidsintervallet utan han får pröva sig fram.

Andra alternativ är att använda faltningar, fouriertransformer och andra signalbehandlingstekniker.

6.2 Svagheter i DRPL

För DRPL ligger svagheten mest i obalans för ledare. Är ledarna parvis perfekt balanserade kommer avvikelser i effektförbrukning vara av storleksordnigen femtojoule per klockcykel med varianser runt $10^{-}29$ kvadratjoule per klockcykel.

Obalans i ledarna leder till avvikelser i effektförbrukning. Typiskt så kommer variansen av effektförbrukningen ändras kvadratiskt med skillnaden i kapacitans mellan ledarna, och effektförbrukningskvoten kommer vara proportionell mot kvoten av konduktanserna, se 0 och 2.

7 Utblick

Det finns en hel del som inte tagits upp i denna artikel och en hel del att utforska.

Aktiva komponenter på ett chip skulle kunna användas för att reglera effektförbrukning.

Man skulle kunna implementera logiken med logiskt ekvivalenta block men som förbrukar olika mycket effekt och sen låta slump avgöra vilken som används.

Något som inte diskuterats i detalj är implementationen av DRPL, det kan göras på många sätt.

A Notation och definitioner

Notationen som används skiljer sig en del från standardnotation inom ingenjörsvärlden och drar mer åt det hållet som används inom teoretisk såväl som matetmatisk fysik. Notationen följer [Fur22c] som läsaren bör konsumerat vid det här laget.

$\mathbb{B}$ används för att beteckna mängden $\{0, 1\}$ som är mycket vanlig inom digital elektronik.

$\oplus$ används för den exklusiva eller-operatorn, som definieras bitvis enligt

$$\begin{aligned} \oplus : \mathbb{B} \times \mathbb{B} &\to \mathbb{B} \\ (\alpha, \beta) &\mapsto \alpha \oplus \beta \\ &= |\alpha - \beta| \end{aligned} \tag{3}$$

Och från den definitionen för längre bitsträngar

$$\begin{aligned} \oplus : \mathbb{B}^d \times \mathbb{B}^d &\to \mathbb{B}^d \\ (\alpha_\mu, \beta_\mu) &\mapsto \alpha_\mu \oplus \beta_\mu \\ &= |\alpha_\mu - \beta_\mu|_\mu \end{aligned} \tag{4}$$

Resultatet blir alltså en bitsträng i vilken varje komponent är resultaten av en $\oplus$ mellan motsvarande komponenter i operanderna.

B Förkunskaper

Läsaren bör ha konsumerat [Fur22c] och [Fur22b] som båda är av förberedande karaktär.

B.0 Naturfilosofi

Artikeln är skriven på ett sätt som gör att den inte går in särskilt mycket i detaljer vad gäller fysiken. Läsaren bör vara någorlunda bekväm inom:

- ⋆ Kretselektronik, se exempelvis [Wes14]
- ⋆ Elektrodynamik, se exempelbis [Fur22a], [Jac62] eller [Nor]

B.1 Matematik

Inga höga krav ställs på matematik för att ta till sig andemeningen i denna artikel, men för att verkligen förstå och kunna gå vidare rekkomenderas att känna sig bekväm med:

- ⋆ Linjär algebra, se exempelvis [Gus13]
- ⋆ Multilinjär algebra, se exempelvis [Gre78]
- ⋆ Reell analys, se exempelvis [Lar05a], [Lar05b], [Kol75], [Spi67] och [Rud66]
- ⋆ Sannolikhetslära, se exempelvis [Kol56] och [Jay79]
- ⋆ Fourieranalys, se exempelvis [Fol09]

C Vidare läsning

Härifrån kan en intresserad läsare givetvis fortsätta till [Fur22d].

Är man intresserad av att fördjupa sig i döljning som ämne så kan man ju alltid läsa de böcker och artiklar som bidragit mest till det här kvädet: [MO07], [Eff10], [Pee13], [Sch04], [Ros97], [al99] och [al09].

Vill man ha en djupare förståelse för fysiken bakom effektförbrukning och elektronik är det rimligt att läsa de böcker som använts för skrivandet av denna artikel: [Jac62], [Fur22a], [Nor], [Gri99], [Che89], [Hec15], [Mag05], [Sch95], [Sch00], [Fol08], [Wei05], [Dre65], [Bae92], [Fur22b], [Kra87], [McK93] och [Wes14].

Vill man lära sig mer om matematiken som används inom döljning och attacker mot döljning så är sannolikhetslära och multilinjär algebra de områden som jag anser skulle ge mest lön för ansträngingen. Inom dessa områden rekommenderar jag följande böcker som även använts inom skrivandet av denna artikel: [Kol56], [Jay79], [Gus13], [Gre78], [Dar94] och [Spi70]

D Slumpgeneratorer

En *slumpgenerator* (*eng*: True Random Number Generator, TRNG) samlar entropi från omvärlden genom att utföra mätningar på till synes oförutsägbara fenomen. Dessa kan vara av många olika slag och en djupgående diskussion är olämpligt för den här artikeln[†], men några exempel kan nämnas: Temperatur, accelerometerdata, elektronikjitter, strålning (α, β, γ)[♯] och fysiologiska

[†]Vad som utgör en sann slumpkälla och vilka fenomen som är lämpliga att mäta är kanske inte helt lätt att definiera respektive avgöra. Definitionsmässigt kokar det ned till huruvida det på förhand går att ställa upp en sannolikhetsfördelning sådan att den förutsäger resultatet av en fysikalisk process. Slump, *dolda variabler* och oförutsägbara fysikaliska processer diskuteras utförligt i [Jay79], [Sch95], [Fol08], [Neu32], [Kol56], [Fur22e], [Mag05], [Sch00], [Wei05], [Bae92], [Dre65], [Fol08], [Sha13], [Gri18], [Dir82], [Hen15] och [Wol94].

[♯]Strålning syftar här på emitterade partiklar från sönderfall, typiskt av atomkärnor. Vid övergångar mellan olika tillstånd kan kärnor emittera partiklar. Typiskt är dessa av sex (eller åtta) slag. Protonemission, neutronemission, elektronemission, positronemission, kärnemission och gammaemission. Till dessa har vi även neutrinoemissioner av två olika slag men de är så svåruppmätta att vi inte rimligtvis kan bygga slumpgeneratorer baserade på dem och dessutom sker de alltid i kombination med positronemission eller elektronemission. Elektron- och positronemission kallas för β-sönderfall (ibland skriver man mer specifikt β^- för

rörelser (som puls och leders vinklar). Ju bättre slump desto högre entropi erhålls. Entropin för en slumpad variabel, x, med utfallsrum, $\{x_0, x_1, ..., x_n\}$, och tillhörande sannolikheter, $\{p_0, p_1, ..., p_n\}$, kan definieras såsom (detta värde kallas för *shannonentropi*):

$$\mathscr{S}(x) = -\sum_{\mu=0}^{n} p_\mu \log_2 p_\mu \tag{5}$$

där ett värde av $\mathscr{S} = 1$ innebär fullständig slump, alltså att varje utfall är lika sannolikt.

Under entropiinsamling är det viktigt att slumpgeneratorn inte låter sig styras av yttre påverkan, därför används sensorer och självtester.

D.0 Diffusiva funktioner

En *diffusiv* funktion, f_{diffusiv}, är en endomorfi över någon mängd, M,

$$f_{\text{diffusiv}} : M \to M \tag{6}$$

som inte nödvändigtvis behöver vara surjektiv eller injektiv. De viktigaste egenskaperna för en diffusiv funktion är att det ska vara svårt att hitta tillbakalyftningen (eller ens ett element i den)

$$\begin{aligned} x &= preim_{f_{\text{diffusiv}}}(y) \\ &\in \wp(M) \end{aligned} \tag{7}$$

för något $y \in M$, och att det inte ska finnas något samband mellan funktionens värde och argument för några par av värden och argument. Med andra ord ska man inte kunna ”invertera”[†] funktionen och ”små ändringar” i argument ska inte nödvändigtvis ge ”små ändringar” i värde[♯].

Funktioner av denna typ kallas ibland *envägsfunktioner*.

elektronemission och β^+ för positronemission). Bland kärnemissioner är helium-4 (alltså en kärna med två protoner och två neutroner) så vanlig att den fått ett eget namn, α-sönderfall. Neutronemission och protonemission kallas ibland för n-sönderfall respektive p-sönderfall fast dessa termer är inte så vanliga. När någon av dessa har skett så är kärnan oftast i ett instabilt tillstånd vilket den rättar till genom att avfyra en eller flera fotoner, och det kallas γ-sönderfall. Av dessa är de rimligaste slumpkällorna α-, β- och γ-sönderfall.

[†]Oegentligt med tanke på att f_{diffusiv} inte ens nödvändigtvis är en bijektion, och därmed kanske inte ens har någon invers. Detta uttryck kommer vanligvis av ett missförstånd eller ihopblandning av termerna invers och tillbakalyftning. Tillbakalyftningen av en funktion, $f(x)$, skrivs ibland, något slarvigt, $f^{-1}(x)$, eller, något mindre slarvigt men fortfarande slarvigt, $f^{-1}[x]$.

[♯]Oegentligt med tanke på att vi inte ens definierat vad en liten ändring är. M behöver inte vara utrustad med någon metrik eller norm. I dessa sammanhang är dock $M = \mathbb{B}^d$ där d är något positivt heltal. Vanliga mått på små förändringar på $\mathbb{B}^d$ mellan två element, v och w skulle exempelvis kunna vara $HD(v, w) = HW(v \oplus w)$ eller $|v - w|$ där elementen har en numerisk tolkning, exempelvis så som är vanligt inom digital elektronik, alltså att $v = v_\mu \in \mathbb{B}^d$ har det numeriska värdet $v = \sum_{\mu=0}^{d-1} 2^\mu v_\mu$

D.1 Hashfunktioner

En speciell kategori av funktioner som ligger mycket nära diffusiva funktioner är *hashfunktioner*. En hashfunktion, $f_{\#}$, är en funktion,

$$f_{\#} : \bigcup_{\mu=0}^{\infty} \mathbb{B}^{\mu} \to \mathbb{B}^{d} \tag{8}$$

som uppvisar de egenskaper två centrala egenskaperna som diffusiva funktioner uppvisar. En hashfunktion tar alltså en godtyckligt lång sträng av bitar, $v \in \bigcup_{\mu=0}^{\infty} \mathbb{B}^{\mu}$, och avbildar på en bitsträng av fix längd (d bitar), $f_{\#}(v) \in \mathbb{B}^{d}$.

Denna typ av funktioner är mycket användbara. En sak de kan användas för är att tilldela en godtyckligt stor mängd data ett (nästan) unikt[†] värde av mer hanterlig storlek som (nästan) inte avslöjar någonting om vilka data som orsakade det. Ett annat användningsområde är som efterbehandling av slump.

Hashfunktioner väljs vanligtvis så att de går att implementera effektivt i hårdvara.

D.2 Utvärdering av slump

När entropin är samlad och slumptalet verkats på med en hashfunktion utvärderas resultatet, så att det inte uppträder några mönster. Om allting går rätt till skall tre (3) krav uppfyllas (åtminstone önskar man komma så nära att uppfylla dem som möjligt):

0. Det ska inte gå för en anfallare, givet kunksap om ett tillstånd, att räkna ut nästa tillstånd
1. Det ska inte gå för en anfallare, givet kunksap om ett tillstånd, att räkna ut föregående tillstånd
2. Slumptalen skall följa en lådformad sannolikhetsfördelning i utfallsrummet

Det sista villkoret kan uttryckas som att varje utfall skall vara lika sannolikt som varje annat. Om exempelvis utfallsrummet för x är $\mathbb{B}^{d}$, för något positivt heltal, d, så har vi totalt 2^{d} olika utfall. Sannolikheten för att få utfallet $x_{\mu} \in \mathbb{B}^{d}$ skulle alltså vara $p_{\mu} = \frac{1}{2^{d}} \in \mathbb{R}$. Detta ska givetvis gälla för varje $\mu \in \{0, 1, \ldots, 2^{d} - 1\}$ där μ syftar på en numrering av alla element i $\mathbb{B}^{d}$.

Denna utvärdering sker vanligen genom att genererad slump körs igenom mjukvara som letar efter mönster och mäter entropi. Det går även att behandla

[†]Det är givetvis inte unikt utan, om hashfunktionen är någorlunda vettigt byggd, det finns oändligt många element ur definitionsmängden som motsvarar varje element ur värdemängden, så den är verkligen inte unik. Emellertid kommer man vara tvungen att leta länge om d är ett stort tal, typ hundra eller nåt.

slumpdata med okulär inspektion, för det har visat sig att människan ännu kan tävla med datorer i att se mönster.

Referenser

[Fur22a] Anders Furufors. *Elektromagnetiska fält, vågledare och antenner, Del I: Elektrodynamikens principer*. 2022.

[Fur22b] Anders Furufors. *Effektanalys, Del I: Effektanalysens principer*. 2022.

[Jac62] David Jackson. *Classical Electrodynamics*. Wiley, 1962.

[Nor] Martin Norgren. *Kompendium i elektromagnetisk fältteori, 2H1250*. Avdelningen för teoretisk elektroteknik, Alfvénlaboratoriet, KTH.

[Fur22c] Anders Furufors. *Matematisk introduktion*. 2022.

[Spi70] Michel Spivak. *A Comprehensive Introduction to Differential Geometry*. Publish or Perish, 1970.

[Sch50] Erwin Schrödinger. *Struktur der Raum-Zeit*. Cambridge Science Classics, 1950.

[Wes14] Westcott och Westcott. *Basic Electronics: Theory and Practice*. Mercury Learning & Information, 2014.

[Gus13] Ivar Gustafsson. *Linjär algebra och numerisk analys*. Institutionen för matematik, Chalmers, 2013.

[Gre78] Werner Greub. *Multilinear Algebra*. Springer Verlag, 1978.

[Lar05a] Arne Persson och Lars-Christer Böiers. *Analys i en variabel*. Studentlitteratur AB, 2005.

[Lar05b] Arne Persson och Lars-Christer Böiers. *Analys i flera variabel*. Studentlitteratur AB, 2005.

[Kol75] Andrej Kolmogorov. *Introductory Real Analysis*. Dover Publications, 1975.

[Spi67] Michel Spivak. *Calculus*. Publish or Perish, 1967.

[Rud66] Walter Rudin. *Real and Complex Analysis*. McGraw-Hill, 1966.

[Kol56] Andrej Kolmogorov. *Foundations of the Theory of Probability*. Martino Fine Books, 1956.

[Jay79] Edwin Thompson Jaynes. *Probability Theory: The Logic of Science*. Cambridge University Press, 1979.

[Fol09] Gerald Folland. *Fourier Analysis and Its Application*. American Mathematical Society, 2009.

[Fur22d] Anders Furufors. *Effektanalys, Del III: Maskering*. 2022.

[MO07] Mangard och Oswald. *Power Analysis Attacks*. Springer Verlag, 2007.

[Eff10] Rankl och Effing. *Smart Card Handbook*. Fourth. Wiley, 2010.

[Pee13] Eric Peeters. *Advanced DPA Theory and Practice*. Fourth. Springer Verlag, 2013.

[Sch04] Helms och Schmidt och Nebel. "Leakage in CMOS Circuits – An Introduction". I: *Springer Verlag* (2004).

[Ros97] Markus Kuhn och Ross Anderson. “Low Cost Attacks on Tamper Resistant Devices”. I: *Springer LNCS* (1997).

[al99] Kocher & al. “Differential Power Analysis”. I: *Cryptography Research, Inc* (1999).

[al09] Danger & al. “Overview of Dual Rail with Precharge Logic Styles to Thwart Implementation-Level Attacks on Hardware Cryptoprocessors”. I: *HAL* (2009).

[Gri99] David Griffith. *Introduction to Electrodynamics*. Cambridge University Press, 1999.

[Che89] David Cheng. *Field and Wave Electromagnetics*. Addison-Wesley, 1989.

[Hec15] Eugene Hecht. *Optics*. Pearson, 2015.

[Mag05] Michele Maggiore. *A Modern Introduction to Quantum Field Theory*. Oxford University Press, 2005.

[Sch95] Peskin och Schroeder. *An Introduction to Quantum Field Theory*. CRC Press, 1995.

[Sch00] Franz Schwabl. *Advanced Quantum Mechanics*. Springer Verlag, 2000.

[Fol08] Gerald Folland. *Quantum Field Theory: A Tourist Guide for Mathematicians*. American Mathematical Society, 2008.

[Wei05] Steven Weinberg. *The Quantum Theory of Fields*. Cambridge University Press, 2005.

[Dre65] Bjorken och Drell. *Relativistic Quantum Fields*. McGraw-Hill, 1965.

[Bae92] John Baez. *Introduction to Algebraic and Constructive Quantum Field Theory*. Princeton University Press, 1992.

[Kra87] Kenneth Krane. *Introductory Nuclear Physics*. Third. Wiley, 1987.

[McK93] John McKelvey. *Solid State Physics for Engineering and Materials Science*. Krieger Pub Co, 1993.

[Dar94] Richard Darling. *Differential Forms and Connections*. Cambride University Press, 1994.

[Neu32] John Neumann. *Mathematische Grundlagen der Quantenmechanik*. Springer Verlag, 1932.

[Fur22e] Anders Furufors. *Kvantinformatik, Del I: Kvantfysikens principer*. 2022.

[Sha13] Ramamurti Shankar. *Principles of Quantum Mechanics*. Springer Verlag, 2013.

[Gri18] David Griffiths. *Introduction to Quantum Mechanics*. Cambridge University Press, 2018.

[Dir82] Paul Dirac. *The Principles of Quantum Mechanics*. Clarendon Press, 1982.

[Hen15] Måns Henningson. *Börja med kvantfysik*. Institutionen för fundamental fysik, Chalmers, 2015.

[Wol94] Haken och Wolf. *The Physics of Atoms and Quanta*. Springer Verlag, 1994.