Hacking af trådløse enheder

Synopsis udført af Mads Kristensen i valgfaget security

02/06-2017 Af Mads Kristensen

Indholdsfortegnelse

02/06-2017 Af Mads Kristensen

Indledning

Min synopsis danner grundlag for min mundtlige eksamen på 4. semester på datamatikeruddannelsen i valgfaget security. Mit valgte specialiseringsområde omhandler hacking af trådløse enheder, som bruger 2.4 GHz protokollen. Jeg har valgt dette emne, da jeg flere gange har overvejet sikkerhedsrisici ved trådløse enheder og brugen af disse. Jeg finder det også interessant, at folk bruger alle disse enheder uden at skænke det en tanke, at de muligvis åbner op for hackerangreb.

Problemstilling

Computere, tablets og mobiltelefoner er en del af manges hverdag, og dertil kommer også et hav af trådløse enheder, som man med lethed kan koble op til dem. Disse enheder bruger ofte 2.4 GHz til at kommunikere eller sende data – men er det sikkert? Jeg vil i dette projekt undersøge sikkerhedsrisici ved disse.

Problemformulering

- Hvilken sikkerhedsrisiko udgør eksterne enheder, som bruger 2.4 GHz protokol?
- Hvad er risikoen specielt ved:
    - Keyboard
    - Mus

- Hvordan udføres angreb gennem disse eksterne enheder i praksis?
    - Generelt overblik
    - Et detaljeret eksempel på et angreb

- Hvordan kan firmaer beskytte sig mod disse angreb?
    - Eksempler på firmaers beskyttelsespolitik

02/06-2017 Af Mads Kristensen

Metode

Jeg vil tilegne mig viden via research på nettet i form af guides og videoer. Jeg vil også undersøge nettet for videnskabelige artikler og forholde mig kildekritisk til disse i en tid, hvor "fake news" er en ting. I den praktiske del vil jeg forsøge at lave en keysniffer via en Raspberry Pi og en 2.4 GHz transceiver. Mit hack vil være et proof of concept og kun muligt, da det er en skoleopgave. Jeg har snakket med min vejleder om de etiske omstændigheder ved at hacke, og som skoleopgave ville dette være OK.

Planlægning

	Mandag	Tirsdag	Onsdag	Torsdag	Fredag
Arbejdsuge 1	Indsamle data om 2.4 GHz protokollen og finde sikkerhedsrisici				Indskrive research i synopsis
Arbejdsuge 2	Indsamle data om angreb af enheder, der bruger 2.4 GHz protokollen				Indskrive research i synopsis
Arbejdsuge 3	Indsamle data om angreb af enheder, der bruger 2.4 GHz protokollen		Indsamle viden om beskyttelse mod angreb		Indskrive research i synopsis
Arbejdsuge 4	Indsamle viden om beskyttelse mod angreb		Indsamle viden om andre trådløse sikkerhedshuller til fremlæggelse		Indskrive research i synopsis og aflevere.

Research

2.4GHz protokollen

I dag kører rigtig mange enheder på 2.4 GHz, da det er et godkendt ISM [1]bånd, som ikke interferere med f.eks. radio eller tv. Grunden til, at denne frekvens er så populær i dag, er, at antennestørrelsen for at udsende frekvenser på 2.4 GHz er minimal i forhold til andre. Det betyder også, at vi ser trådløse telefoner, Bluetooth-teknologi, bilalarmer, drone controllere osv., som kører på dette ISM bånd. Trådløse dongles til keyboards og mus er ikke underlagt regler om sikkerhed, som f.eks. Bluetooth er, hvilket betyder, at producenterne af trådløse dongles skal sørge for at kryptere deres produkter (Herman, 2017).

Sikkerhedsrisici ved 2.4 GHz

Den største sikkerhedsbrist i trådløse mus og keyboards ligger i alle de dongles, som følger med dem. Herunder beskriver jeg nogle forskellige metoder, hvorpå man kan udnytte usikre dongles.

Keysniffing og injection:

Keyboard og mus, som ikke sender enkrypterede pakker via det trådløse signal, kan være ofre for keysniffing eller injection. Der er også mange ældre modeller, som har forældet firmware, som bruger en svag enkryptering, som er forholdsvis let at hacke. Risikoen ved dette kan være, at hackeren kan læse alt, hvad et udvidende offer skriver. Hackeren kan også sende uønskede kommandoer til offerets computer ved at sende falske keystrokes, som i princippet kan gøre skade. Hackeren er ofte nødt til at være tæt på offerets computer for, at et angreb som dette ville lykkes. Der er dog lavet forsøg med at gøre det fra ca. 200 meters afstand (Greenberg, 2017). Dette forsøg blev lavet af Marc Newlin (Newlin, 2017), som også var skaberen af MouseJack, som en reaktion på, at Lenovo udtalte, at man skulle være tæt på computeren, som man ville hacke.

Deaktivering af enheder:

Hackere kan sende specielle pakker til ofres computere, som kan deaktivere ofrets trådløse enheder. Risikoen er ikke specielt høj ved et sådant angreb, men er mest bare irriterende for et offer.

[1] ISM (Industrial, Scientific and Medical Band) er frekvensbånd, der er typegodkendt eller individuelt godkendt til at sende og modtage udstyr og kan benyttes licensfrit. (Wikipedia)

Forced pairing:

Forced pairing går ud på, at en hacker tvinger en dongle til at danne par med et keyboard, som ikke er ofrets. Dette kan lade sig gøre, fordi Logitech bl.a. bruger flere enheder til samme dongle, og dermed tillader en pairing mode session. Hvis et sådant angreb lykkes, da vil ofrets computer kunne lide stor skade, da hackeren kan downloade virus eller andet malware direkte ned på ofrets computer.

Dette skema har jeg valgt at sætte op for at give et bedre overblik over sikkerhedsrisici over beskrevne trådløse hacks. Påvirkning og sandsynlighed bliver bedømt på en skala fra 1 til 5, hvor 1 er lav og 5 er høj.

Sikkerhedsrisiko	Påvirkning (P)	Sandsynlighed (S)	Sikkerhedsrisikoscore (P x S)	Løsninger
Keysniffing og injection	4	2	8(lav)	• Producenten af enhederne skal lave en software opdatering • Sikre, at uvedkommende ikke kommer for tæt på offerets computer • Fjerne dongle, når man forlader computeren
Deaktivering af enheder	2	2	4(lav)	• Sikre, at uvedkommende ikke kommer for tæt på offerets computer • Fjerne dongle, når man forlader computeren
Forced pairing	5	4	20(høj)	• Producenten skal sørge for, at hackere ikke kan sende pairing signaler f.eks. med encryption • Sikre, at uvedkommende ikke kommer for tæt på offerets computer • Fjerne dongle, når man forlader computeren

Alle disse angreb er med lidt kodefærdigheder og lidt googling forholdsholdvis nemme at genskabe. Internettet hjælper både med guides til opsætning og endda også brudstykker af kode. Jeg har anført både Newlin, Kamkar og Goodspeed i min bibliografi, da disse personer åbent fortæller om, hvordan man gør. Newlin og Kamkar linker bl.a. til deres Github projekter.

Sikkerhedsrisici ved keyboards:

Keyboards er nemme ofre ved et keysniffing eller injection angreb, da man som beskrevet før, kan læse keystrokes, injecte falske keystrokes eller skifte et offers keyboard ud med sit eget. Det er den mest udbredte form for sniffing af trådløse enheder, da man kan få mest information ud af dette. Dette vil blive uddybet under MouseJack Attack.

Sikkerhedsrisici ved mus:

I dag findes der mus med intern hukommelse, hvor man f.eks. kan gemme en makro ved specifikke klik på dem. Dette kan hackere udnytte, da man kan injecte ondsindede makroer og få ofrets computer til at køre diverse hacks. Det kan være en hackers vej ind i et offers computer, som så kan medføre skade. Mus behøver ikke have en intern hukommelse for, at en hacker kan læse klik og bevægelser, men kan derimod danne en gateway til ofrets computer via den dongle, som musen bruger. Mus er er nogle gange dårligere krypteret end keyboards, hvilket MouseJack (Newlin, 2017) angrebet også udnytter. Dette vil blive uddybet under MouseJack Attack.

Udførsel af angreb i praksis

Der er mange måder at udføre angreb mod trådløse enheder på. Det første man skal sikre sig er, at ofrets enhed er modtagelig overfor angreb. Det betyder, at ofret skal bruge en enhed, som sender pakker via 2.4 GHz signalet. Det næste er, at man skal anskaffe sig en receiver til at modtage sådanne signaler, og det kan man let anskaffe sig ved hjælp af lidt internetshopping. Den mest udbredte receiver, som bliver brugt ved et trådløst angreb, er NRF24L01, som er lavet til arduino. Under et angreb vil hackeren sidde og lytte på frekvenser i luften, indtil ofret rykker på sin mus eller skriver noget på sit keyboard. Nu kan hackeren identificere ofrets enhed, og hackeren kan nu vælge, hvilket angreb han vil udføre. Angrebene er beskrevet ovenfor, og hackeren vil som oftest lægge en virus evt. noget ransomware, overføre filer eller udnytte adgangen til ofrets computer. Det er i hvert fald den tendens, som man ser i det nuværende hackermiljø. Jeg fandt en artikel fra en blog, der beskriver, hvad man kan forvente af hackerangreb i 2017 (Kovacs, 2017).

02/06-2017 Af Mads Kristensen

Jeg vil på uddannelsesmæssig baggrund herunder beskrive mit eget forsøg med at keysniffe via en NRF24L01 antenne. Forsøget er baseret på Samy Kamkars KeySweeper (Kamkar, 2017), som jeg har forsøgt at lave min egen version af.

Materialer:

- Raspberry Pi
- NRF24L01 transceiver
- Trådløst keyboard

Keysniffing af et Microsoft trådløst keyboard:

Jeg har forsøgt at lave en 2.4 GHz keysniffer ved hjælp af min Raspberry Pi. Det er derfor ikke lykkes mig, da jeg har haft svært ved at læse/finde signalet fra mit trådløse keyboard.

Måden, hvorpå jeg har forsøgt at læse fra min NRF24L01 tranceiver, har været forskellige. Jeg startede med at installere et bibliotek, som lå online og hed RF24. Biblioteket findes i mange forskellige udgaver og er lavet til arduino. RF24 biblioteket kan sagtens bruges til Raspberry Pi, da der er lavet support til python 3 i det. Problemet for mig var, at RF24 (TMRh20, 2017) biblioteket er skrevet til at sende og modtage signaler fra en NRF24L01 til en anden. Herunder ses de forskellige filer, som biblioteket kan køre på en Linux platform:

```
pi@raspberrypi:~/test/rf24libs/RF24/examples_linux $ ls
extra                          interrupts         pingpair_dyn.py
gettingstarted                 Makefile           readme.md
gettingstarted_call_response   Makefile.examples  transfer
gettingstarted_call_response.cpp pingpair_dyn     transfer.cpp
gettingstarted.cpp             pingpair_dyn.cpp
```

Jeg har forsøgt at køre gettingstarted et par gange, men det er kun lykkedes mig at få nogle reads fra min Raspberry Pi pins en gang ud af halvtreds forsøg. Herunder ses et fejlslagent forsøg:

```
pi@raspberrypi:~/test/rf24libs/RF24/examples_linux $ sudo ./gettingstarted
RF24/examples/GettingStarted/
STATUS           = 0x00 RX_DR=0 TX_DS=0 MAX_RT=0 RX_P_NO=0 TX_FULL=0
RX_ADDR_P0-1     = 0x0000000000 0x0000000000
RX_ADDR_P2-5     = 0x00 0x00 0x00 0x00
TX_ADDR          = 0x0000000000
RX_PW_P0-6       = 0x00 0x00 0x00 0x00 0x00 0x00
EN_AA            = 0x00
EN_RXADDR        = 0x00
RF_CH            = 0x00
RF_SETUP         = 0x00
CONFIG           = 0x00
DYNPD/FEATURE    = 0x00 0x00
Data Rate        = 1MBPS
Model            = nRF24L01
CRC Length       = Disabled
PA Power         = PA_MIN

 ************ Role Setup ***********
Choose a role: Enter 0 for pong_back, 1 for ping_out (CTRL+C to exit)
>
```

Det første man ser er et read af de pins, som Raspberry Pien regner med at læse data fra. I alle tilfælde får vi 0x00, hvilket betyder, at den tror, at der ikke er ledninger sat til de rigtige steder, som vist herunder:

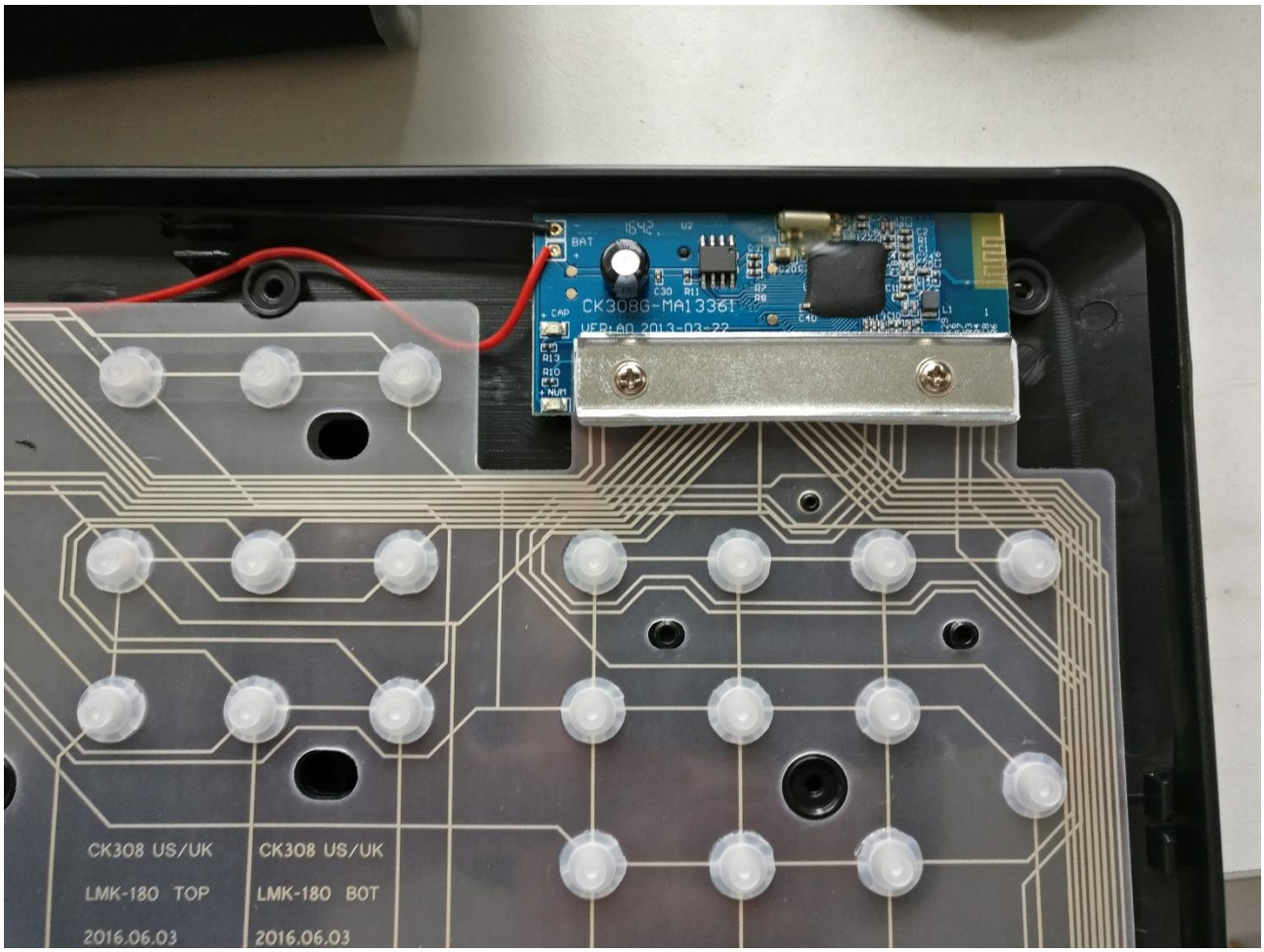
```
 ************ Role Setup ***********
Choose a role: Enter 0 for pong_back, 1 for ping_out (CTRL+C to exit)
>1
Role: Ping Out, starting transmission

Now sending...
RF24 HARDWARE FAIL: Radio not responding, verify pin connections, wiring, etc.
RF24 HARDWARE FAIL: Radio not responding, verify pin connections, wiring, etc.
RF24 HARDWARE FAIL: Radio not responding, verify pin connections, wiring, etc.
RF24 HARDWARE FAIL: Radio not responding, verify pin connections, wiring, etc.
RF24 HARDWARE FAIL: Radio not responding, verify pin connections, wiring, etc.
RF24 HARDWARE FAIL: Radio not responding, verify pin connections, wiring, etc.
RF24 HARDWARE FAIL: Radio not responding, verify pin connections, wiring, etc.
RF24 HARDWARE FAIL: Radio not responding, verify pin connections, wiring, etc.
RF24 HARDWARE FAIL: Radio not responding, verify pin connections, wiring, etc.
RF24 HARDWARE FAIL: Radio not responding, verify pin connections, wiring, etc.
RF24 HARDWARE FAIL: Radio not responding, verify pin connections, wiring, etc.
RF24 HARDWARE FAIL: Radio not responding, verify pin connections, wiring, etc.
RF24 HARDWARE FAIL: Radio not responding, verify pin connections, wiring, etc.
```

Jeg har flere gange sikret mig, at alle ledninger sad på de rigtige pins, så jeg gik lidt i stå med forsøget her og valgte derfor at fokusere på den teoretiske del i stedet.

Idéen bag min keysniffer var at opsnappe pakker sendt trådløst fra et keyboard via en NRF24L01 tranceiver. Jeg vil herunder forklare, hvordan det var meningen, at det skulle virke ved hjælp af Samy Kamkars KeySweeper projekt (Kamkar, 2017).

Det første man gør er at skille sit trådløse keyboard ad og tjekke chippen, der sidder i. Her kan man se, hvilken type chip, der sidder i, og i mit tilfælde var det en Ck308g. Det modelnr. slår man op på nettet for at tjekke, hvilken frekvens det trådløse signal kører på. I mit tilfælde var det 2.408 – 2.474 GHz[2]. Herunder ses chippen fra det keyboard jeg brugte i mit forsøg:



NRF24L01 tranceiveren er beregnet til at kommunikere frem og tilbage mellem to enheder og har derfor brug for at kende en MAC adresse. MAC adressen skal bruges, så vi kan kommunikere med keyboardet eller i det her tilfælde keysniffe.

[2] https://fccid.io/RTX-CK308G

Det næste er, at man skal lure keyboardets MAC adresse, og det er lidt tricky, men man kan snyde sig til den ved hjælp af NFR24L01. Først må man erfare, hvordan keyboardet sender pakker ud. Nedenfor er vist, hvordan et keyboards packet kan se ud.

Preamble 1 byte	Address 3-5 byte	Packet Control Field 9 bit	Payload 0-32 byte	CRC 1-2 byte

Preamblen fortæller, at der er en pakke på vej, så modtageren ved, at pakken er parat til at ankomme. Adressen fortæller, hvem pakken, der bliver afsendt, er til. Der er nogle gange et Packet Control Field, som indeholder informationer omkring, hvordan modtageren skal agere. Payloaden er den data, vi ser, da resten bliver fjernet. Men som hacker er det vigtigt at kende MAC Adressen, da det er den, vi skal bruge for at bruge for at keysniffe. Det sidste, der bliver sendt, er en CRC[3], som er en checksum, der checker om dataen er korrekt, men det svarer ikke til dataintegritet. Hvis ikke CRC matcher resten af pakken, da vil pakken blive droppet.

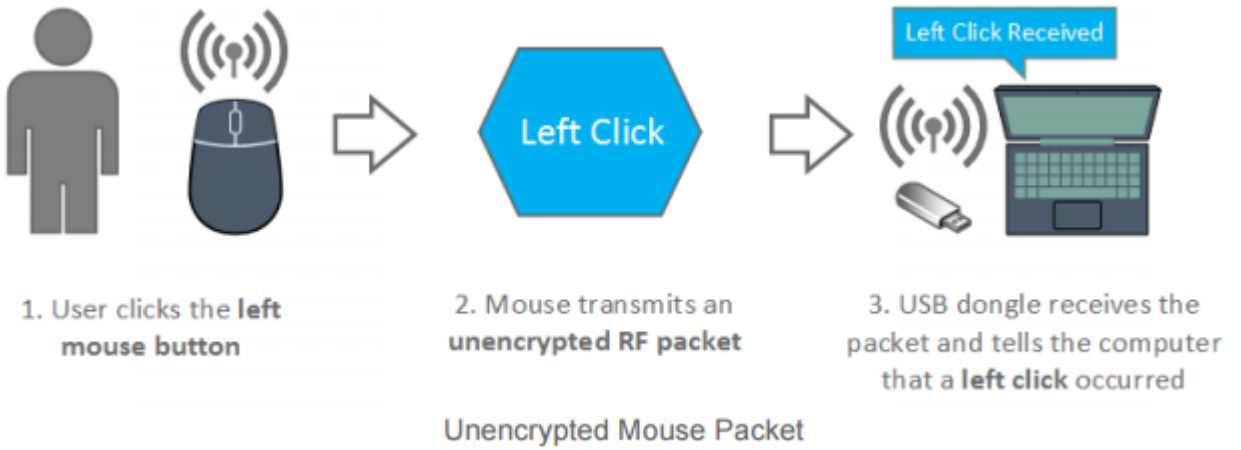
Hvordan finder vi så vores MAC adresse?

Den gode mand Travis Goodspeed (Goodspeed, 2017) har hjulpet os med dette. Han fandt ud af, at hvis man sender MAC adressen, som preamble til keyboardet, så kan man forvirre radiosignalet. Når man sender MAC adressen til radioen i keyboardet, da vil radioen sende payloaden tilbage. Forskellen fra en normal kommunikation til nu er, at payloaden nu også indeholder MAC adressen fra det keyboard, som man vil keysniffe. Travis Goodspeed fandt ud af at i stedet for at sætte MAC adressen til 3-5 bytes, så kan man sætte den til 2 bytes, som teknisk set er ugyldigt. Checksummen og CRC vil ikke matche og give en ugyldig pakke, men CRC kan deaktiveres, og så bliver pakken sendt uanset hvad. Så ved hjælp af Travis Goodspeed kan man nu modtage pakker sendt fra et keyboard.

MouseJack attack

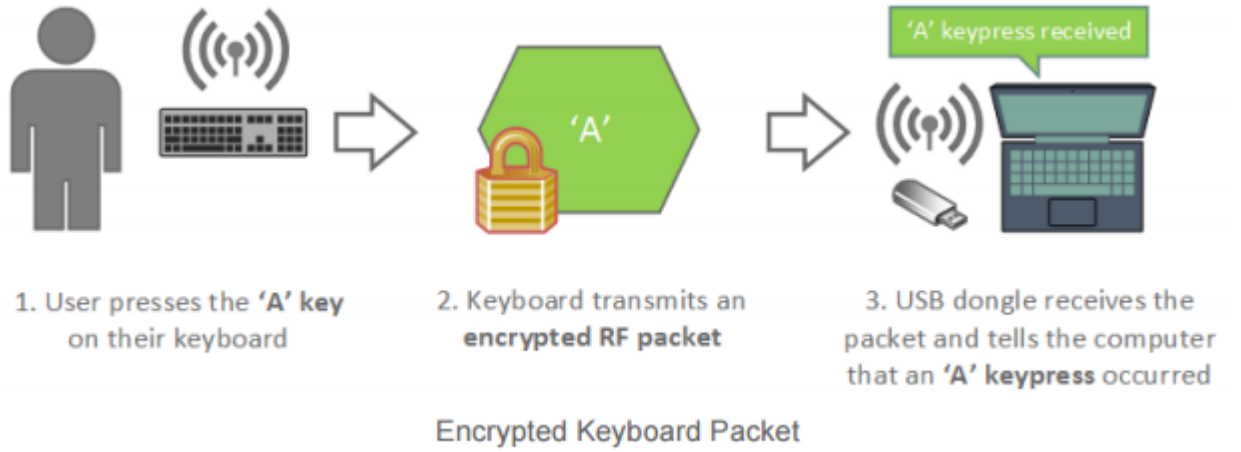
MouseJack angrebet udnytter, at mange trådløse mus ikke krypterer deres pakker, som mange trådløse keyboards gør. Herunder er vist, hvordan en mus sender en pakke, som ikke er krypteret:

[3] CRC(Cyclic Redundancy Check) er en ikke sikker hashfunktion, som checker for om den data, der er sendt stemmer overens med udregnede CRC byte.(Wikipedia)

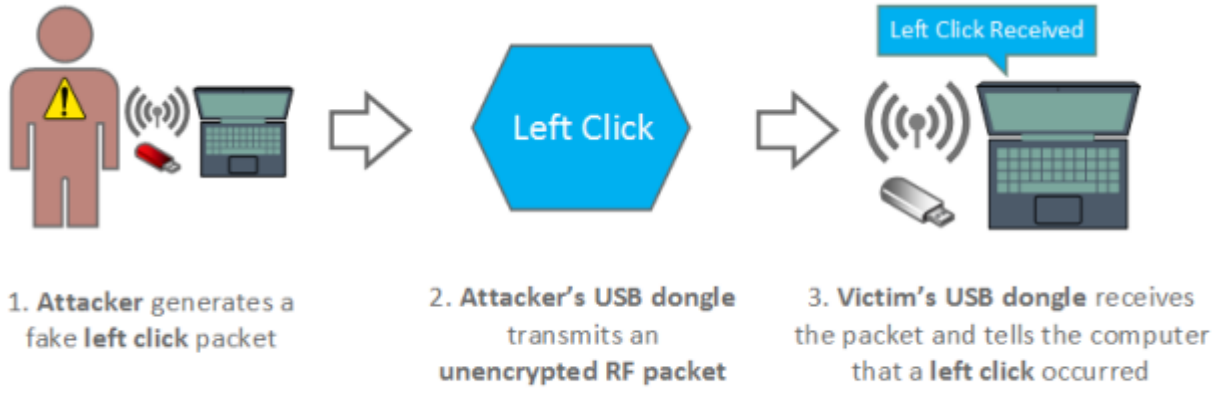


Dette var tilfældet ved alle de mus, som blev testet ved Marc Newlins forsøg (Newlin, 2017).

Trådløse keyboards vil som oftest sende pakkerne krypteret, fordi producenterne gerne vil forhindre, at hackere lytter på deres pakker. De gør det således:



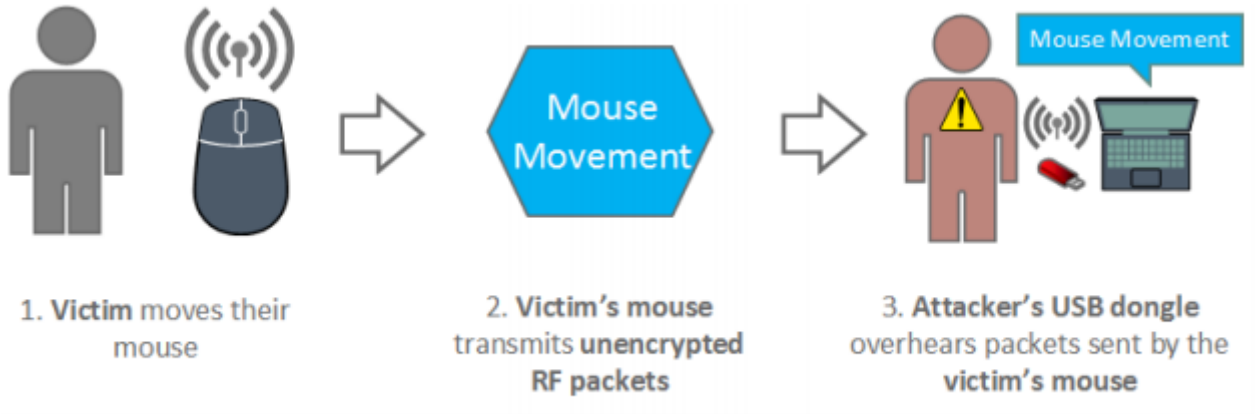
Grunden til, at angrebet hedder MouseJack, er netop, at man som hacker kan udnytte måden, hvorpå musen modtager pakker. Man kan som hacker sende falske pakker og få musen til at reagere derefter. Her er et eksempel på hvordan:



Spoofed Unencrypted Keyboard Packet

I stedet for at generere et venstre klik, så er der en fejl i måden, hvorpå musens dongle modtager pakker, som kan udnyttes til at sende et tastetryk på et keyboard i stedet.

MouseJack angrebet fungerer lidt på samme måde, som Kamkars KeySweeper (Kamkar, 2017). Først vil man sidde og lytte efter pakker, som er sendt afsted fra ofrets computer.



Attacker Identifying a Victim's Mouse or Keyboard

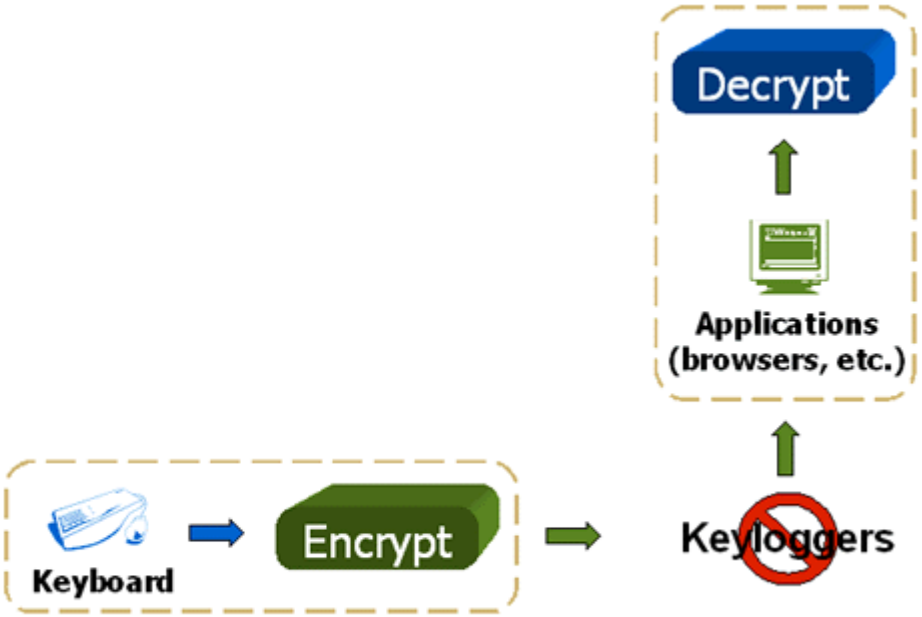
Når hackeren har opsnappet en pakke, så kan han udføre de forskellige angreb, som er beskrevet længere oppe f.eks. en forced pairing.

MouseJack angrebet udnytter svagheden i den sensor, som jeg købte til min keysniffer, og bruger en anden type dongle, der hedder Crazyradio PA USB dongle. Denne dongle er egentligt lavet til droner, men kan nemt udnytte packet sniffing ved nogle linjer python kode (Newlin, 2017).

Hvordan kan firmaer beskytte sig mod disse angreb?

Firmaer/privatpersoner kan i sig selv ikke gøre særlig meget andet end at smide de gamle keyboard, som de har ud, og så købe nogle nye. Gamle keyboards har ikke mulighed for at opdatere sin firmware, og det er heri, at problemet ligger. Producenterne kan dermed gøre en del ved dette problem, men det er ikke alle producenter, som gør noget ved problemet. Jeg har i min research erfaret, at wired.com har været i kontakt med en del af de producenter, som MouseJack angrebet kan ramme. En Logitech talsmand har blandt andet udtalt "The vulnerability would be complex to replicate and would require physical proximity to the target," og tilføjer "It is therefore a difficult and unlikely path of attack.". Lenovo har sagt, at alvorligheden ved angrebet er "low", og at man skal være tæt på (indenfor 10 meter). Kreatørerne af MouseJack attack har dog herefter været ude og sige, at de har formået at lave injections fra op til 180 meters afstand. Efter MouseJack kom frem, da har Logitech, Lenovo, Dell og Microsoft alle erkendt, at det er en reel sårbarhed og har brugt MouseJack til at lave en sikkerhedsopdatering.

En anden måde, som firmaer kan beskytte sig på, er f.eks. via ekstern software, der krypterer de anslag, som man laver på sit keyboard. Fandt et link med et softwareprogram, der gør netop dette, se figur herunder (qfxsoftware, 2017):



Programmet når altså at enkryptere tastetryk inden en keylogger vil kunne læse dem, bagefter keyloggeren har sniffet, så vil programmet så dekryptere f.eks. i en browser.

Jeg har ikke fundet nogle deciderede firma politikker omkring sikkerhed på dette område, men jeg hæfter mig ved ordene som Sam Kamkar har udtalt ”If you can go wired, Go wired”, som burde være praksis i alle firmaer (Greenberg, 2017).

Konklusion

I min synopsis har jeg berørt sikkerhedsrisici ved trådløse enheder, som bruger 2.4 GHz eller rettere sagt bruger en specifik del af det ISM bånd, som kører på 2.4 GHz. I min research erfarede jeg, at rigtig mange trådløse enheder, mere specifikt mus og tastaturer, udgør en vis form for sikkerhedsrisici. Sikkerhedsrisicien ligger i, at når en person sætter en dongle i til sin trådløse enhed, så har hackere mulighed for at angribe din computer eller bare logge dine tastetryk. Mulighederne for angreb kan være keysniffing eller key injection, hvor hackeren lytter på alle tastetryk eller, at hackeren sender ondsindede pakker mod ofrets computer. Forced pairing er også en mulighed, hvor hackeren ender med at danne par sammen med ofrets dongle, og får dermed kontrol over maskinen. Det er en mere direkte måde, da hackeren ikke behøver at dekryptere hver eneste pakke, da han selv trykker på tasterne. Hackeren kan simpelthen også vælge at deaktivere ofrets enhed, hvilket bare er et irritationsmoment. Af disse tre angreb har jeg vurderet forced pairing til at have den største skadelighed. Ved hjælp af lidt googling, så kan man selv komme rigtig langt med lide kodefærdigheder.Angrebene kan både skade privat personer, men også et firma at blive hacket på den måde. Mus er især sårbare, hvis man kigger på MouseJack angrebet, fordi alle de mus, som de testede, var der ingen kryptering på. Sådan forholder det sig også med keyboard, men her producenterne væsentlig bedre til at sørge for enkryption.

Goodspeed lagde grundstenene for at bruge billige radiotranceivere til at opsnappe pakker sendt fra trådløse enheder. Den information brugte både Newlins MouseJack og Kamkars KeySweeper netop til at finde MAC adressen på den specifikke enhed, der skulle hackes. Det lyder nemt, men det kræver, at man snyder den trådløse enhed til at sende MAC adressen. Det, man gør, er, at man sender MAC adressen til enheden og normalt ville den kun sende sin payload tilbage, men sender nu også sin egen MAC adresse. Når hackeren har MAC adressen, så skal han bare dekryptere de forskellige payloads, som lyttes på.

Mit eget produkt gik i vasken, da jeg ikke kunne få de reads fra den sensor, som jeg ville bruge til at lave min keysniffer med. Det er super ærgerligt ikke at få det til at lykkedes, men det er jo på den anden side også ulovligt. Det har været svært at finde nogle sikkerhedspolitikker om emnet, da det selvfølgelig ikke er noget firmaerne vil have liggende offenligt. Men efter lidt research, så ville jeg foreslå at de kunne bruge en keyscrambler, som dekrypterer hvert anslag på et tastatur.

Min endelige bemærkning må jeg igen lade gå til Samy Kamkar - ”If you can go wired, Go wired”.

Refleksion

Efter jeg har færddiggjort arbejdet med min synopsis, da har jeg tænkt en del over nogle forskellige ting som har undret mig.

Da jeg researcher på 2.4 GHz protokollen finder jeg langsomt ud af, at ja det er rigtig fint at det licensfrit, men der er ingen styring af sikkerhed ligesom der er i bluetooth. Det her er et problem, da alle producenter vel er glade for at der er licensfrie ISM bånd, men lidt glemmer at de selv skal opfinde noget kryptering hvis de bruger det. Nu har jeg ikke tal på hvor mange enheder, der ikke er enkrypterede, men der må være en del, som i rigtig rigtg mange. Nu har jeg jo primært fokuseret på mus og tastatur, men jeg overvejede da også og jeg kunne hacke mine trådløse høretelefoner, og så bare skifte musikken. I dag er der alt for mange ting, som kører på den her protokol, og IoT(Internet of Things) stormer jo frem. Jeg tænker bare hvorfor er der intet fokus på det her?

I forhold til min arbejdsproces, da har jeg måske ikke helt fulgt mit planlægninsskema til punkt og prikke. Jeg tænkte faktisk først lidt over slet ikke og have det med, da jeg synes det var åndssvagt. Jeg er klogere nu, selvfølgelig, og det er jeg, fordi jeg faktisk har støttet mig ubevidst til de ting jeg havde skrevet deri. Jeg har ikke fulgt det særlig godt, men det gav mig et fantastisk overblik, igen ubevidst overblik. Det er helt klart noget jeg vil huske til næste opgave jeg skal skrive.

Jeg kan rigtig godt lide synopsis formatet, men jeg havde alt for lidt tid. Havde jeg haft længere tid, så havde jeg med sikkerhed fået lavet mit produkt, og hvem ved måske har jeg en ny fritidsbeskæftigelse med det i et par uger?

Bibliografi

Goodspeed, T. (19. Maj 2017). *travisgoodspeed.blogspot.dk*. Hentet fra http://travisgoodspeed.blogspot.dk/2011/02/promiscuity-is-nrf24l01s-duty.html

Greenberg, A. (23. Maj 2017). *Wired.com*. Hentet fra https://www.wired.com/2016/02/flaws-in-wireless-mice-and-keyboards-let-hackers-type-on-your-pc/

Herman, J. (10. Maj 2017). *Wired.com*. Hentet fra https://www.wired.com/2010/09/wireless-explainer/

Kamkar, S. (19. Maj 2017). *samy.pl*. Hentet fra http://samy.pl/keysweeper/

Kovacs, N. (30. Maj 2017). *community.norton.com*. Hentet fra https://community.norton.com/en/blogs/norton-protection-blog/top-ten-cyber-security-predictions-2017

Newlin, M. (10. Maj 2017). *bastille.net*. Hentet fra https://www.bastille.net/research/vulnerabilities/mousejack/technical-details

qfxsoftware. (1. Juni 2017). *qfxsoftware.com*. Hentet fra https://www.qfxsoftware.com/ks-windows/how-it-works.htm

TMRh20. (30. Maj 2017). *http://tmrh20.github.io*. Hentet fra http://tmrh20.github.io/RF24/