La radio qui venait du froid

SSTIC 2014

Alain <ozwald> SCHNEIDER

alain.schneider<AT>cogiceo.com
@OzwaldFR

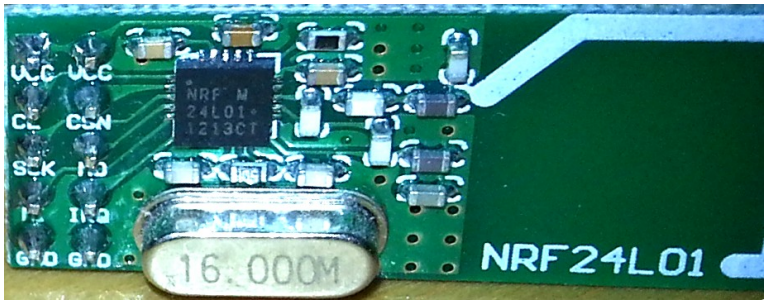
COGICEO

Un jour, ce clavier fît son apparition sur le bureau de l'un de mes collègues.

C'est un clavier sans fil...L'occasion idéale pour vérifier si j'ai raison de n'acheter que des périphériques filaires !

Cette présentation courte vise à résumer les recherches que j'ai effectué jusqu'à présent, et à vous lister des pistes alléchantes.

En me renseignant, j'ai découvert rapidement l'existence des puces nRF24L01+



C'est un composant « sur étagère » du constructeur *Nordic Semiconductor* qui :

- est adressable en SPI
- offre une communication radio bidirectionnelle
- utilise la bande des 2.4GHz
- consomme peu d'électricité
- module son signal en GFSK
- est probablement présent dans votre souris sans fil :-)

Comment écouter les communications de ces puces ?

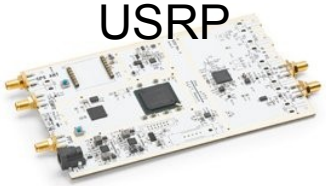
Les idées immédiates :

RTLSDR

Radio logicielle
Disponible à faible coût (~20$)

Ne peux pas analyser les fréquences supérieures à 2GHz

USRP



Radio logicielle
Émission et réception
De quelques MHz à 6GHz

600$

HackRF

Radio logicielle
Émission et réception
De quelques MHz à 6GHz

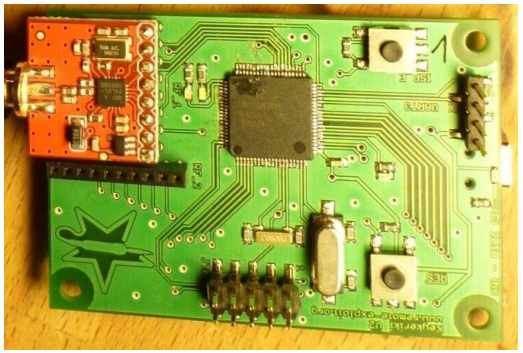
300$
Pré-commande seulement

Aucune bonne idée...Quid de l'état de l'art ?

Comment écouter les communications de ces puces ?

Une recherche de l'état de l'art fait ressurgir du passé une présentation à CanSecWest 2010

Practical Exploitation
of
Modern Wireless Devices



Thorsten Schroeder & Max Moser

Les communications sont écoutables :

- Avec du matériel accessible (~200$)
- Dans un contexte embarqué
- Grâce à quelques concessions

Les leçons :

- Une méthode d'écoute
- Un matériel fonctionnel
- Quelques surprises chez certains périphériques

Résumé de la méthode « Keykeriki »

Pour comprendre la méthode d'écoute, attardons-nous sur le fonctionnement de ces puces :

- La communication se fait par paquets « [Enhanced] Shockburst »
- Chaque puce peut émettre et recevoir des paquets
- Les communication peuvent se faire à 2Mb/s, 1Mb/s, et 250kb/s
- Chaque paquet est précédé d'un préambule radio (alternance de 8 bits à 0 et 1)
- Chaque paquet envoyé l'est à une adresse unique, d'une taille variable entre 3 et 5 octets
- La charge utile peut être de taille variable, entre 0 et 32 octets
- Un CRC facultatif peut être placé en fin de paquet

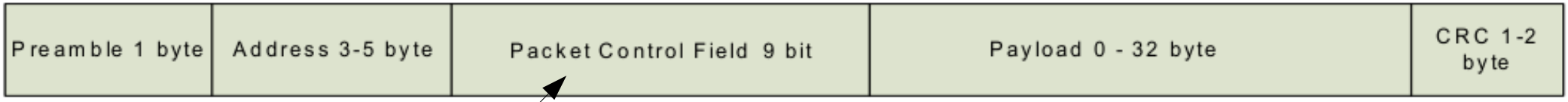
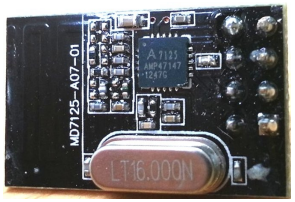


Figure 5. An Enhanced ShockBurst™ packet with payload (0-32 bytes)

P.I.T.A.

Résumé de la méthode « Keykeriki »

Keykeriki s'articule autour de trois composants principaux :



3€

Une puce A7125

Son rôle est de démoduler, en GFSK, un flux continu de bits sur l'un des canaux utilisables par les nRF24L01+

~60€

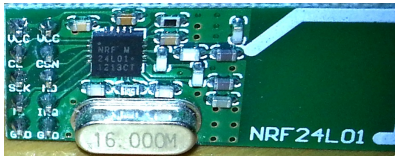
Un processeur ARM à 100MHz

Son rôle est double :

- configurer les puces radios
- **identifier, dans le flux à 2Mb/s en provenance de l'A7125, les paquets nRF24L01+ valides**

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

Figure 5. An Enhanced ShockBurst™ packet with payload (0-32 bytes)



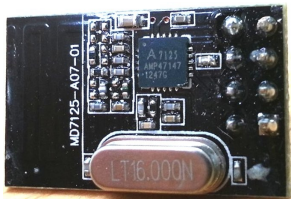
3€

Une (ou plusieurs) puce nrf24L01+

Une fois les paramètres de communication identifiés, ces puces sont configurées à l'identique ce qui permet une suite d'interception fiable.

Ça marche ! Malheureusement, un « Keykeriki » reste encore trop cher pour une simple blague.

« Keykeriki » moins cher

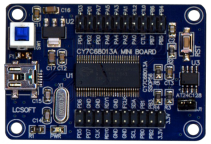


3€

Une puce A7125

Son rôle est de démoduler, en GFSK, un flux continu de bits sur l'un des canaux utilisables par les nRF24L01+

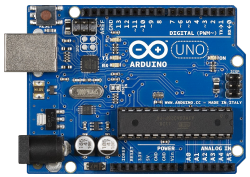
Remplace l'ARM



~6€

Un analyseur logique contrefait

Son rôle est de transférer le flux à 2Mb/s jusqu'à un PC



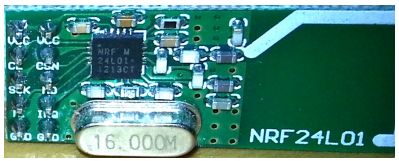
~7€

Un arduino

Son rôle est de configurer les puces radio

Un PC

Son rôle est d'identifier les paquets valides dans le flux brut



3€

Une (ou plusieurs) puce nrf24L01+

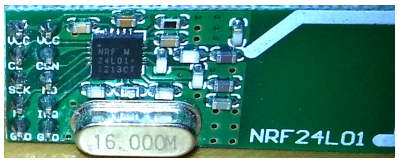
Une fois les paramètres de communication identifiés, ces puces sont configurées à l'identique ce qui permet une suite d'interception fiable.

Le code source sera open source sur : https://github.com/cogiceo

COGICEO

Plus simple, la méthode « T.Goodspeed »

La méthode de Travis s'articule autour de trois composants principaux :



Une puce nrf24L01+

3€

Son rôle est d'écouter en mode « promiscuous », puis en mode ciblé.

Un goodfet

~40€

Servant d'interface entre la puce nRF24L01+ et un ordinateur.

Un ordinateur

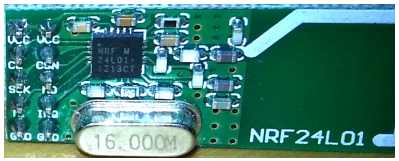
Son rôle est de configurer la puce radio (via le goodfet) et d'identifier les adresses valides dans le flux capturé en mode « promiscuous »

Pour utiliser la puce nRF24L01+ en mode « promiscuous » à 2Mb/s une astuce est nécessaire :

Interprétation légitime	bruit			préambule	adresse					Suite du paquet « Enhanced shockburst » : en-tête, charge utile, CRC	bruit		
Flux à 2Mbps	xx	55	00	55	01	02	03	04	05	YY...YY	xx	xx	xx
Interprétation de Travis	bruit	préambule	adresse		Charge utile de taille maximale d'un paquet Shockburst								

Plus simple, la méthode « T.Goodspeed »

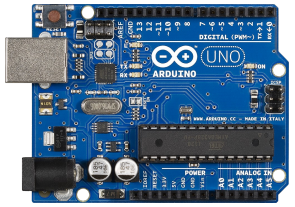
N'ayant pas de goodfet, j'ai ré-implémenté l'attaque avec un arduino.



3€

Une puce nrf24L01+

Son rôle est d'écouter en mode « promiscuous », puis en mode ciblé.



~7€

Un arduino

Servant d'interface entre la puce nRF24L01+ et un ordinateur.

Un ordinateur

Son rôle est de configurer la puce radio (via le goodfet) et d'identifier les adresses valides dans le flux capturé en mode « promiscuous »

Le logiciel est ~~erade~~ OpenSource (BSD & GPLV2) : https://github.com/cogiceo

Plus simple, la méthode « T.Goodspeed »

A la première itération de la méthode basée sur arduino[1], là où Travis obtenait des résultats parfaits[2] ...

```
94 22 e1 10 8d 45 49 51 32 84 80 62 a5 43 92 9d 6c 94 2e 8a c4 17 53 05 32 aa 31 84 2f 6b 12 55
01 02 03 02 01 10 17 ff ff 00 00 08 0f 00 00 7a 69 ff ff 91 c0 d1 3b ee a3 ef 56 60 17 88 87 50
6f 12 2a aa 5a db aa bd 16 94 6a d5 5a a5 8b a9 5a a3 0a 92 29 57 22 55 ab 69 eb c5 b7 ea b6 ca

48 80 40 c2 08 53 04 6b 17 81 6c d0 28 c1 18 55 0b 6b 38 82 29 30 d5 a8 c3 01 48 18 66 04 34 8c
01 02 03 02 01 10 17 ff 00 00 00 08 34 00 00 7a 69 ff ff 3a 69 18 2d 23 6d 6b fd 46 59 63 35 32
40 54 04 55 4a 64 1c 20 90 62 11 11 19 00 00 10 0d 77 01 27 a8 b3 42 00 26 14 40 04 85 00 ca ad

b4 ca dd 7a f5 37 b5 2f cb ea a3 64 4d 8a a8 55 a6 b5 54 b2 54 6e dd da b3 55 6d 55 26 da a9 23
01 02 03 02 01 10 17 ff aa 00 00 08 3e 00 00 7a 69 ff ff e8 ca 47 36 59 3d 7e fa 49 6d 9b 27 5e
42 00 92 2a 88 15 54 4a 25 67 e5 6d 5d aa 5a ad d4 1b 16 bd 25 08 b4 ad 53 ad 4e 48 ab 5a 68 d5
```

...je me suis retrouvé avec un joyeux bouillon :

```
87 46 A6 54 1A 29 29 66 49 0C C9 97 55 48 A4 81 00 29 12 88 AA BB 7B DF B6 AA FE 5E CB AD 5B 7D
9A AA AA AA AA AA AA AA AA AA AA 55 01 02 03 02 01 CC 5A 4B 82 80 67 96 6A A9 55 C2 37 D2 CC 2A
84 44 12 AA 28 49 19 04 A4 51 15 01 7A A7 44 A2 42 82 41 24 08 40 88 20 41 10 40 A9 44 02 04 5C

41 3A FD AB AA BB 25 4E 2A AA FA 25 5F 55 D6 B7 7A FB A7 6B FB AA CB EB 2A 14 CA AA A9 56 49 C
00 00 00 00 00 00 00 55 01 02 03 02 01 CC 1C 0D 02 80 46 1D 48 58 4E 56 BA 0E 49 5F 0E D5 AA CD
A0 40 A9 0A 50 41 0A CC 95 5B 2C 15 10 96 AA AA 93 2D 9A EA B6 D6 69 2A 68 BA 36 95 6D 52 A9 5A

D1 55 45 B5 53 31 D2 29 53 55 11 05 54 2C AA 53 24 54 93 56 C1 5B 4D 22 51 12 AE D5 22 CA 4E C5
AA AA AA AA AA AA AA AA AA AA AA 55 01 02 03 02 01 CD 5F 9C 02 00 4D 96 DA 74 4B 4A AA D6 59 40
8A A9 46 D0 A5 48 52 08 34 5A AD D2 A5 94 75 2A 99 6C AF CB ED ED AB BB 9A D2 F9 76 5F A2 AB FE
```

Leçon à retenir : l'endianness, c'est important.

1 : en vente à environ 10€ dans toutes les bonnes crèmeries
2 : http://pastebin.com/8CbxHzJ9

J'arrive à intercepter les communication radio du clavier que je ciblais, MAIS, mon collègue a déménagé à 3 bureaux de là...il est trop loin[1]. Comment augmenter la portée ?

Le 21 janvier 2014[2], « cyberexplorer » publie sur son blog une méthode d'écoute des communication utilisant un dongle RTLSDR. Pour rappel, les RTLSDR s'arrêtent, au mieux, à 2GHz, comment peut-il écouter des communications à 2.4GHz ?...

Un MMDS Downconverter (et son alimentation)

Son rôle est de remoduler les signaux de 2.4GHz à environ 400MHz

Un RTLSDR

Son rôle est de démoduler les signaux GFSK ramené à 400MHz et de les transmettre à l'ordinateur.

Un ordinateur

Son rôle est de configurer le RTLSDR et d'identifier les adresses valides dans le flux capturé.

A l'heure actuelle je ne suis pas parvenu à reproduire ses résultats. Plus d'infos dès que j'ai avancé :)
Dernier rebondissement : mon alimentation était en panne.

1 : l'histoire prouvera que, finalement, il n'étais pas trop loin.
2 : à peine quelques jours avant la cloture du CFP SSTIC...

Bon, on parvient à intercepter les charges utiles des puces nRF24L01+ en provenance de nos claviers et souris. Mais est-ce grave ?

Si les charges utiles sont chiffrées, ce n'est pas trop grave :

- Logitech communique sur un chiffrement « AES ».
- Quid de Microsoft ?

C	0A	78	06	01	**C2**	**98**	76	0A	C0	**C8**	98	35	0A	C0	CD	**5B**
K					**CD**	**98**	**35**	**0A**	**C0**	CD	98	35	0A	C0	CD	
P	0A	78	06	01	**0F**	**00**	**43**	**00**	**00**	**05**	**00**	**00**	**00**	**00**	**00**	
	Device type	Packet type	Model	?	Sequence ID		Flags/Meta			HID Code						Checksum

(Key-Down) Packet with device address

CD 98 35 0A C0

En résumé :

- On ne connaît pas d'attaque publique sur les claviers Logitech (AES)
- Les souris et « touches spéciales » de clavier ne sont pas chiffrées...
- Les communications de claviers Microsoft étaient chiffrées avec un XOR...en 2010 !
- Nous avons proposé deux ré-implémentations* de méthodes connues pour en réduire le coût jusqu'à approcher celui d'une pinte de bière.

Quelques pistes :

- Peut-on facilement injecter des commandes ?
- Peut-on r00ter en n'utilisant que la souris et/ou les « touches spéciales » ?
- Quid des « petits constructeurs » (Rainbow, etc.)
- **A quelle distance peut-on écouter ces communication ?**
- **Qu'en est il des claviers Microsoft en 2014 ?**

* : https://github.com/cogiceo

QUESTIONS / DEMO