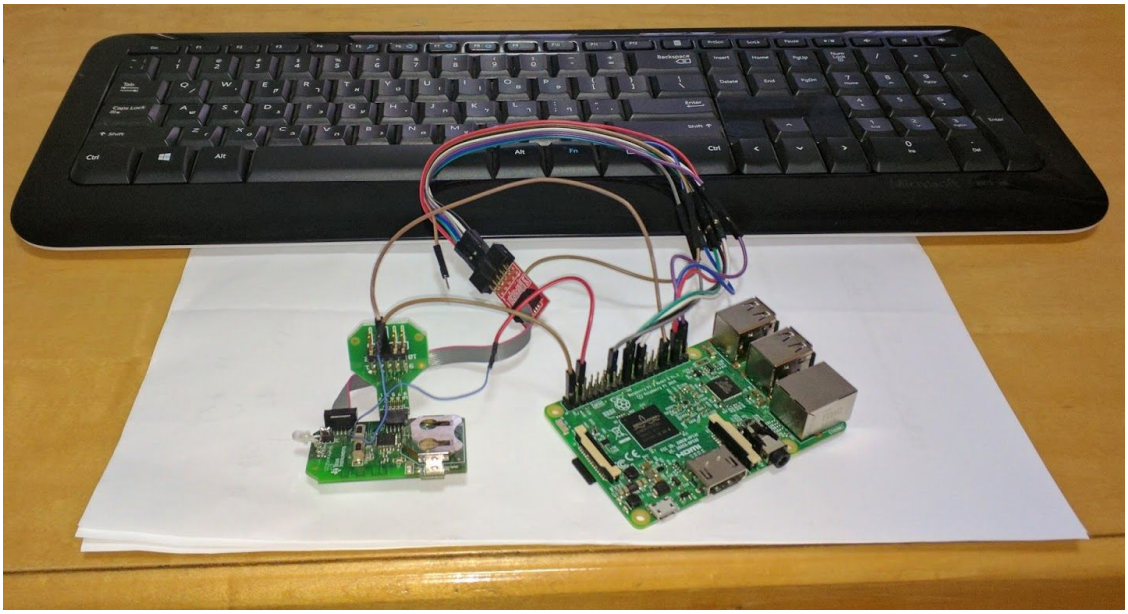
Advanced Computer Systems - Wireless Keylogger

By Rony Jacobson and Misha Seltzer

The project

We're using TI's cc2541 chip to wirelessly detect, sniff and store wireless keyboard's keystrokes for later evaluation.
Wireless keyboards are very widely used, but many models (most non-BT models) are not encrypted at all, and with a cheap HW (CC2541 evaluation board costs $99, and a non standard version (HM-11) can be found for as little as 12$ or even less) can be easily sniffed.



The inspiration

A company named “Bastille” that is in the business of securing IoT devices published a whitepaper with a huge list of such keyboards and listed their protocols and naive encryption very well.
All that was left for us to do is to scan the “airspace”, and log keystrokes.

The Demo

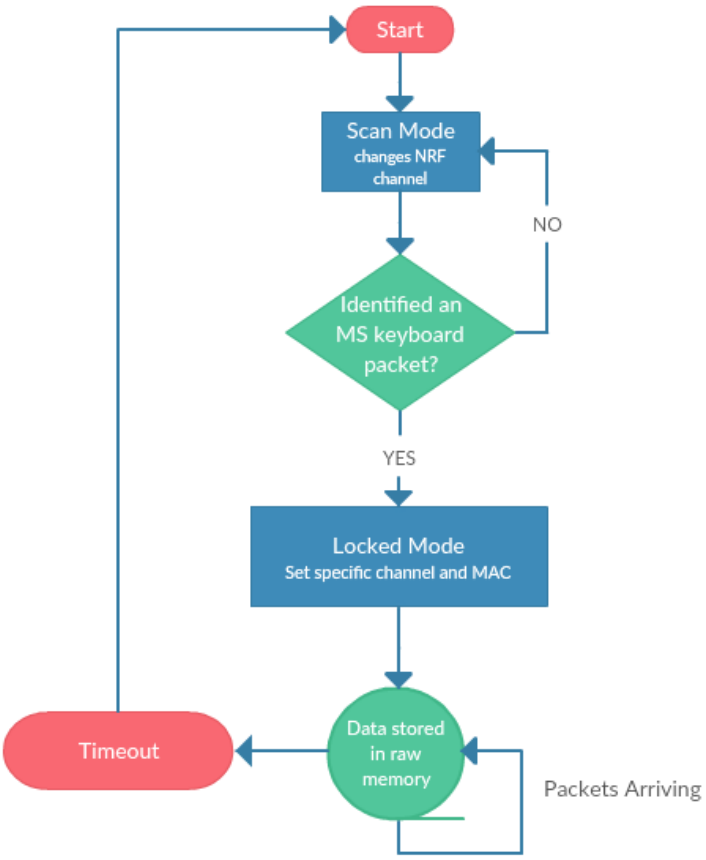
For simplicity, our demo targets only unencrypted Microsoft keyboards over the NRF protocol.

Logic

Our logic is broken into two main pieces:

The keylogger

The keylogger hardware (based on the cc2541 evaluation board) starts its work in scan mode. In this mode we’re scanning the “airspace” in promiscuous mode and waits until it finds a packet that seems to resemble a Microsoft keyboard (the checksum seems to fit the data, and the “mac address” has the expected prefix and suffix).



While in scan mode, the keylogger changes its NRF channel every second (there are <80 possible channels on which Microsoft keyboards can communicate, we scan all 80). Once such a packet is found, the keylogger moves to a “locked” mode, where we set the cc2541 to listen for a specific mac address on a specific channel. Every packet received is stored in its raw form in memory, and the “data” reset line is turned on.
After a long timeout of inactivity in “locked” mode, the keylogger switches back to “scan” mode.

The receiver

Our receiver is a Raspberry Pi device, that connects via SPI to the keylogger, and, assuming the “data” reset pin is HIGH, pulls keystrokes from the memory and produces them into text using a decryption function adjusted to the MS keyboard protocol.

Disclosure

We’ve never gotten to the fully autonomic mode of work, thus in our demo the keylogger does not move from “scan” to “locked” modes and does not change channels automatically. The keylogger in our demo is connected to the receiver (the Raspberry Pi) continuously, and the latter sends the “Change Channel” and “Change Mode” commands.