PROXMARK III

GETTING STARTED GUIDE FOR UBUNTU (GNOME) USERS

Version 1

INTENDED AUDIENCE

The Proxmark III is intended for users that are either competent hardware or software developers (preferably both). Users that do not understand the basic principles behind RFID may have difficulty using the device.

The Proxmark III is a RFID development tool. Typically, an "out of the box" proxmark3 with the latest firmware can run acquisitions in LF and HF mode, output traces, decode a number of different RFID credentials and do some operations in ISO 15693 and ISO 14443 a and b modes. If you really want to get the most out of this device, you will need to start enhancing the firmware yourself to suit your needs.

INTRODUCTION

The proxmark3 is a powerful general purpose RFID tool, the size of a deck of cards, designed to snoop, listen and emulate everything from Low Frequency (125 kHz) to High Frequency (13.56 MHz) tags.

This device can do almost anything involving almost any kind of low (125 kHz) or high (13.56 MHz) frequency RFID tag. It can act as a reader. It can eavesdrop on a transaction between another reader and a tag. It can analyse the signal received over the air more closely, for example to perform an attack in which we derive information from the tag's instantaneous power consumption. It can pretend to be a tag itself. It is also capable of some less obviously useful operations that might come in handy for development work.

HISTORY

"The Proxmark III is a device developed by Jonathan Westhues that enables sniffing, reading and cloning of RFID (Radio Frequency Identification) tags. For my master thesis I wanted to look at the communication of Mifare Classic cards. Mifare Classic is used in many applications and is the most popular contactless card around. It is used in e-ticketing, public transport and access control. The higher-level protocol is kept secret by the manufacturer (NXP). I made an implementation of the ISO14443 type A standard for the Proxmark since Mifare is based on this communication standard.

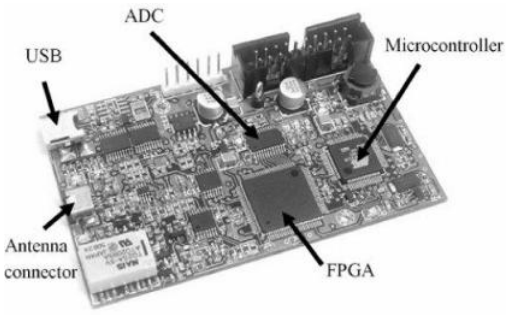
After a lot of debugging and many noisy traces the Proxmark was ready for some real analysis. I focused on the Mifare Classic card and was happy to let the communication between card and reader appear on my screen. I could see the anticollision phase where the reader selects the card to communicate with. This was followed by an authentication and after that all communication was encrypted. The findings of this research are published on arxiv.org as A Practical Attack on the Mifare Classic

In December 2007 I saw the presentation of Karsten Nohl and Henryk Plotz where they showed how they reverse engineered the Mifare Classic chip. I was working on the same subject in Nijmegen. The Mifare system relied on security by obscurity and now the secrets are revealed there is no card-level security left. A video on YouTube shows a demo that we gave on March 12th. It shows how we access a building with a cloned card.

We will not release the software used in the MIFARE Classic attacks.

Gerhard de Koning Gans, Roel Verdult"

HARDWARE OVERVIEW



ADC (ANALOG TO DIGITAL CONVERTER)

The analogue signal that comes from the antenna circuit is fed into an 8-bit Analogue to Digital Converter (ADC). This delivers 8 output bits in parallel which represent the current voltage retrieved from the field.

FIELD PROGRAMMABLE GATE ARRAY

The 8 output pins from the ADC are connected to 8 pins of the Field Programmable Gate Array (FPGA). An FPGA has a great advantage over a normal microcontroller in the sense that it emulates hardware. A hardware description can be compiled and flashed into an FPGA.

Because basic arithmetic functions can be performed fast and in parallel by an FPGA it is faster than an implementation on a normal microcontroller. Only a real hardware implementation would be faster but this lacks the flexibility of an FPGA.

The FPGA can therefore be seen as dynamic hardware. It is possible to make a hardware design and flash it into the memory of the FPGA. This gives some major advantages:

- "Hardware" errors can be corrected; the FPGA can be flashed with a new hardware design.
- Although not as fast as a real hardware implementation, an FPGA is faster than its equivalent on a microprocessor. That is, it is specialized for one job.

The FPGA has two main tasks. The first task is to demodulate the signal received from the ADC and relay this as a digital encoded signal to the ARM. Depending on the task this might be the demodulation of a

100% Amplitude Shift Keying (ASK) signal from the reader or the load modulation of a card. The encoding schemes used to communicate the signal to the ARM are Modified Miller for the reader and Manchester encoding for the card signal.

The second task is to modulate an encoded signal that is received from the ARM into the field of the antenna. This can be both the encoding of reader messages or card messages. For reader messages the FPGA generates an electromagnetic field on power hi and drops the amplitude for short periods.

MICROCONTROLLER

The microcontroller is responsible for the protocol management. It receives the digital encoded signals from the FPGA and decodes them. The decoded signals can just be copied to a buffer in the EEPROM memory. Additionally, an answer to the received message can be send by encoding a reply and communicating this to the FPGA.

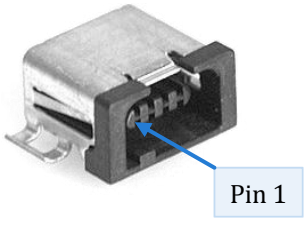
The microcontroller (ARM) implements the transport layer. First it decodes the samples received from the FPGA. These samples are stored in a Direct Memory Access (DMA) buffer. The samples are binary sequences that represent whether the signal was high or low. The software on the ARM tries to decode these samples. When the Proxmark is in sniffing mode this is done for both the Manchester and Modified Miller at the same time. Whenever one of the decoding procedures returns a valid message, this message is stored in another buffer (BigBuf) and both decoding procedures are set to an un-synced state. The BigBuf is limited to the available memory on the ARM. The current firmware has 2 KB of memory reserved for traces (Besides the traces the buffer also stores some temporary data that is needed in the processing). When the BigBuf buffer is full the function normally returns. A new function call from the client is needed to download the BigBuf contents to the computer. The BigBuf is especially useful for protocol investigation. Every single message is stored in this buffer. When a card is emulated or when the Proxmark is used as a reader the BigBuf can be used to store status messages or protocol exceptions.

USB

The USB interface interconnects the Proxmark with an external power supply and / or a computer for advanced functionality.

ANTENNA CONNECTOR

The antenna connector is a Hirose Electric low profile, surface mount, right-angle 4 pin connector. This connector allows for the simultaneous connection of a low and high frequency antenna.



- The low frequency antenna (125/134 kHz) connects to Pin 1 (TP5) and Pin 2 (TP2).
- The high frequency antenna (13.56MHz) connects to Pin 3 (TP4) and Pin 4 (TP3).

GETTING STARTED

It is assumed that the reader of this document is running Ubuntu 11.10. This document is intended as a guide only. Group policies and custom configurations are outside the scope of this document.

A high and low frequency antenna will be required when testing the Proxmark. The antenna construction is outside the scope of this document. It is assumed that the antennas have been assembled and tested prior to reading "Testing the Proxmark".

REQUIREMENTS

- A computer running Ubuntu 11.10 with an available USB port.
  - This guide will most likely work with:
  - Ubuntu 11.04, 10.10 and 10.04.
  - BackTrack 5 R1 and 4 R2.
- A USB Mini-B lead.
- A Proxmark III.
- HF and / or LF antenna for the Proxmark.

A technical understanding of the Proxmark III is not required for the installation process.

DEVELOPMENT ENVIRONMENT INSTALLATION

1. Open a terminal: `CTRL` + `ALT` + `T` .
2. Download (and install) TortoiseSVN, p7zip, and components essential to build the Proxmark from soruce: "`sudo apt-get install subversion p7zip build-essential libreadline5 libreadline-dev libusb-0.1-4 libusb-dev libqt4-dev perl pkg-config wget`".
3. Check out the latest revision of the Proxmark project: "`svn checkout http://proxmark3.googlecode.com/svn/trunk pm3`".
4. Get devkitARM release 32 (4.5.1) from SourceForge: http://sourceforge.net/projects/devkitpro/files/devkitARM/previous/devkitARM_r32-i686-linux.tar.bz2/download
5. Extract the contents of the .tar.bz2: "`tar jxvf devkitARM_r32-i686-linux.tar.bz2`".
6. Create a directory for the arm dev kit: "`sudo mkdir /opt/devkitpro/`".
7. Move the ARM developer kit to the newly created directory: "`sudo mv devkitARM /opt/devkitpro/`".
8. Add the appropriate environment variable: "`export PATH=${PATH}:/opt/devkitpro/devkitARM/bin/`".
9. Add the environment variable to your profile: "`echo 'PATH=${PATH}:/opt/devkitpro/devkitARM/bin/' >> ~/.bashrc`".

PROXMARK DRIVER INSTALLATION

No drivers are required to use the Proxmark within Linux. You will however need to run the client as root.

TESTING THE PROXMARK

You are now at the stage where you should be able to communicate with your Proxmark.

1. Go in to your Proxmark project folder and run the Proxmark software – "`sudo ./client/proxmark3.exe`".
   1.1. You should see something like this:

```
Connected units:
        1. SN: ChangeMe [bus-0/\\.\libusb0-0001--0x9ac4-0x4b8f]
proxmark3>
```

2. Next, check what firmware you are running – "`hw version`".
   2.1. You should see something like this:

```
#db# Prox/RFID mark3 RFID instrument
#db# bootrom: svn 486-suspect 2011-07-18 12:48:52
#db# os: svn 486-suspect 2011-07-18 12:48:57
#db# FPGA image built on 2009/12/ 8 at  8: 3:54
```

3. Connect your antenna(s) to the Proxmark and type in "`hw tune`".
   3.1. You should see something like this:

```
#db# Measuring antenna characteristics, please wait.

# LF antenna: 33.17 V @   125.00 kHz
# LF antenna: 41.89 V @   134.00 kHz
# LF optimal: 41.76 V @   127.66 kHz
# HF antenna:  7.28 V @    13.56 MHz
```

4. Type "`quit`" to exit out of the program.

NOTES

There are many commands available within the Proxmark client. Type "`help`" to list the commands available to you. You get a list of following subcommands by typing in the command you're interested followed by help (Eg. "`hf help`"). For detailed help please read the Proxmark User Manual.

Command shortcutting is permitted. For instance typing "`hf mf re`" will achieve the same results as typing "`hf mf restore1k`" because it is the only "`hf mf`" command available that begins with "`re`".

You might have noticed when you executed the command "`hw version`" there was a "`-unclean`" or "`-suspect`" in the "bootrom" or "os" version information. This information indicates that the code may have changes in the local code versus the subversion revision.

- A clean build is a build that has no local changes versus the subversion revision.
- Unclean builds have local changes versus the subversion revision.
- Suspect builds cannot be compared against the subversion revision.

NEXT STEPS

From here on in the rest is up to you. You might want to confirm that the firmware you're running is the latest. In addition to this you might want to begin reading through the Proxmark source code and making changes of your own.

We suggest you read the following documentation:

- Compiling Proxmark source and firmware upgrading
- Proxmark User Manual
- Antenna Construction Guide

THE FORUM

We hope you thoroughly enjoy using the Proxmark and encourage you to join the forum and introduce yourself - http://proxmark.org/forum/index.php. There are a number of active members willing to assist newcomers. Please note the forum is not a help desk for inexperienced developers. Poor English,

USEFUL LINKS

The Proxmark forum . . . . . http://proxmark.org/forum/index.php

Proxmark project downloads http://code.google.com/p/proxmark3/downloads/list

The Proxmark project http://code.google.com/p/proxmark3/

The Proxmark wiki . . . . . http://code.google.com/p/proxmark3/wiki/HomePage?tm=6

Proxmark repository http://proxmark3.googlecode.com/svn/trunk/

7-Zip http://www.7-zip.org

Tortoise SVN . . . . . http://tortoisesvn.net

Document corrections or comments . . . . . http://proxmark.org/forum/viewforum.php?id=27