Step By Step Method of Programming Multiple Profiles in a SIM Card

F. N. Uchenna[1], A. M. Aibinu[2,3], R.O. Olayiwola[1], E. Abba[3]A. P. Adedigba[2], and T. A. Folorunso[2]
Department of Mathematics[1], Department of Mechatronics Engineering[2], Centre for Open Distance and e-Learing
Federal University of Technology, Minna Nigeria
P. M. B. 65 Minna, Niger State, Nigeria.
E-mail: abiodun.aibinu@futminna.edu.ng; maibinu@gmail.com

Abstract – In this paper, step by step approach to programming multiply profiles in a blank programmable SIM card using python is presented. This paper also introduces SIM card technology and a basic configuration of an ad-hoc network in a Linux server. Hence, the paper provides how N-Profiles can be co-located on a single card in order to solve the problem of using more than one Phones or SIM cards.

Keyword *– Card reader, DHCP, Linux server, Python, PySim SIM card.*

I. INTRODUCTION

The inherent cryptographic information of Subscriber Identification Module (SIM) card helps protects against damage, cloning, attack etc. SIM card cloning among other attacks on SIM cards has encouraged SIM security [4-8]. The smart card on which the SIM is realized is based on microprocessors specifically designed to be tamper-resistant, comprising of a RAM, EEPROM and ROM [2,4]. The ROM is the firmware that stores the Operating System (OS) and SIM applications, the EEPROM stores the phone book, settings, patches, and the RAM is for use by the SIM's OS and applications. A SIM runs a very simple OS that loads up Java Card, a version of the Java Virtual Machine (VM) for smart cards [4]. It essentially runs small Java applets, and each applet is encapsulated and firewalled by the Java VM, preventing sensitive data from leaking to other apps. The phone interacts with the apps through the SIM Application Toolkit (STK) to display information on the screen [10].

Figure 1: SIM Card Sizes

The first SIM card was introduced around 1991 and since then, there have been a steady increase in the number and capabilities of SIM cards [4]. Over the years, SIM cards have been further miniaturized. First, was the full-size SIMs which was followed by mini SIMs, then micro-SIMs, and nano-SIMs [2,11]. These SIMs are also made to be embedded in devices. However, the most desired changes of SIM cards used worldwide is from Full size to Mini SIM card [2]. The Full sized, Mini SIM and Micro SIM cards have the same thickness, but their length and width are reduced as shown in Figure 1.

There has also been an increase in SIM Cards memory sizeranges from 32KB versions to 128KB versions[3,11]. Recently, a 1MB SIM card was launched in order to cater for the increasing memory capacity requirements demanded by new third-generation (3G)phones [2]. The additional memory in the 1MB SIM can be used to store hundreds of multimedia files. Meanwhile the performance of the processor has also improved with the communication speed between the card and the mobile phone almost 40 times faster than a normal 2G SIM card [5]. The SuperSIM card was developed by Oberthur Card Systems and STMicroelectronics for TIM, an Italian mobile operator. Regular SIM cards can store critical subscriber authentication information, as well as text-based data such as phone numbers or SMSs [10].

Problems associated with recent increase in the number of mobile phone uses placed much demand on wireless communication services. Some of the resulting associated problems include poor Quality of Service (QoS), inefficient communication that leads to high rate of call drops [1].Both the Mobile Network Operators (MNO) and the users have tried in their various means to provide solutions but to no avail. MNOs have resorted to increased Base Transceiver Stations (BTS), this has led to increase electromagnetic radiations. On the part of subscribers, the use of multiple MNO, multiple

mobile phones and multiple SIM cards were the solutions provided. However, this has increased the cost of wireless communication services [1].

Thus, in this paper, we present a step-by-step approach to SIM programming with the aim of writing multiple MNO profiles into a Super-SIM thereby limiting the need of subscribers to possess multiple SIM cards or phones.

The remaining part of this paper is organized as follows:In section II, the installations and configuration guide for a SIM card is presented while section III shows how a blank SuperSIM could be programmed with multiple profiles using python and the paper ends with a conclusion.

II. INSTALLATIONS AND CONFIGURATION

The installation of the required software on the system on which the SIM programming software will reside and the step by step configuration of an ad-hoc network is presented in this section.

1. SYSTEM SOFTWARE INSTALLATION

Installation of versions 2.7.x and version 3.6.x of python was undertaken, though some of the codes were later written in python 2.7.15 and will also be ported to 3.6.5. Pyscard, a python package used to read information from the SIM-card via the card reader and initiate communication between the SIM card and the computer via the SIM card reader was installed. Also Pysim, a small command line utility written in python used for programming various programmable SIM-card and also write new profiles into the SIM `card on a computer that runs a Linux server was also installed.

2. Ad-Hoc NETWORK CONFIGURATION

Profiles were written into the SuperSIM card. Each profile has a distinct set of Service Set Identifications and passphrase to allow for direct communication between the profiles in the SuperSIM and the wireless network with no inherent relaying.

An Independent Basic service set (ad-hoc) was configured to allow the profiles written into the SuperSIM to communicate wirelessly without a central controller. The configuration in this project is for a Linux server (if using a windows server, configurations and terms used here may/will differ).

The procedure taken consists of four major steps namely:

1. The network configuration
2. Configuration of the wireless network card
3. Configuration of DHCP server
4. Configure the firewall for iptables masquerading.
5.

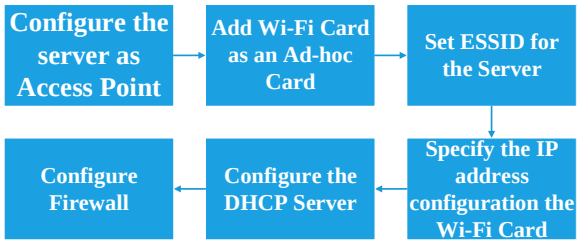

Figure 2: Block diagram showing the work through of the configuration ad-hoc Network

The detail description of the steps taken for the configuration of the ad-hoc network is presented in Figure 2 and subsequently discussed herewith

Step 1. THE NETWORK CONFIGURATION

In order to configure the server as an access point, two networks configurations are needed. One of which should offer the server Internet access (the internet interface). This can be direct Internet access or access through a Network Address Translation (NAT) router –in this work a laptop was used since it has an EthernetInterface (eth0) to access the Internet.

Apart from the Ethernet interface, a wireless interface is required in configuring the server as an access point. The wireless interface is represented as “wl0” for the server used, this may differ in a case where windows or other Linux distributions’ server are to be used. The Internet interface can be configured with most convenient tool. Manual configurations usually are not needed. But for the wireless interface, there is work to be done.

Step 2. CONFIGURATION OF THE WIRELESS NETWORK CARD

The *iwconfig* command is used to configure the wireless card on the command line, before putting the configuration into configuration files, and the is first used to put the Wi-Fi card in ad-hoc mode, using the following command

iwconfig wl0 mode Ad-Hoc

The Wi-Fi card will allow other computers to make a connection only if it is configured in ad-hoc mode. The computer needs an Extended Service Set Identification (ESSID) – a unique name that is used to identify the network and also used as part of the information contained in the profiles coded into the

programmable SIM card (SuperSIM). The following command will set an ESSID for the computer:

iwconfig wl0 essid moes1

iwconfig command is used to verify that the Wi-Fi card have all of the required parameters. Plate 1 is the output of the command:

```
[root@admin ~]#iwconfig

wl0    IEEE 802.11abgn  ESSID:"moes1"
       Mode:Ad-Hoc  Frequency:2.327
GHz  Cell: 96:1E:76:FA:FE:A0
       Tx-Power=15 dBm
       Retry  long limit:7   RTS
thr:off   Fragment thr:off
       Encryption key:off
       Power Management:off
```

Plate I: Output of the *iwconfig*command

The IP address configuration for the Wi-Fi card is specified using the *ip address* command, as follows:

ip address add dev wl0 192.168.78.1 netmask 255.255.255.0

An IP network address that is completely unique is used to specify the IP address in this paper. The computer already knows the default gateway or DNS server from the internet interface, hence, no need to specify one. After specifying the IP address, the Wi-Fi card is up and could now be seen as a wireless access point from other computers in the vicinity.

Step 3. CONFIGURING THE DHCP SERVER

DHCP server is installed to ensure that the access point is allocating IP addresses, using dnf install dhcp. A configuration file with the name /etc/dhcp/dhcpd.conf is made and with the following contents as shown in plate II

```
[root@prisonerofchrist ~]# cat
/etc/dhcp/dhcpd.conf

option domain-name "moes1";

option domain-name-servers 192.168.1.1;

default-lease-time 600;

max-lease-time 7200;

authoritative;

log-facility local7;

subnet 192.168.100.0 netmask 255.255.255.0 {

range 192.168.100.10 192.168.100.20;

option routers 192.168.78.1;

}
```

Plate II: Content of the DHCP configuration file

The DHCP server is started and enabled to reboot automatically. The following command is used to start the server.

service dhcpd start

chkconfigdhcpd on

Step 4. CONFIGURING THE FIREWALL

Final stage of the configuration shows how the NAT configured on the server. This is done using the powerful internal Linux firewall iptables.

NAT is enabled on the server, by running the following commands

iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE

The most important part in this command is the option -o eth0. This option specifies the name of the network interface that is used to connect to the Internet. It was insured that it reflects the correct interface name.

The configuration file /etc/sysconfig/iptables-config is changed, since Fedora and derivatives have a nice feature of saving iptables lines that are typed on the command line if it is given the following three parameters in iptables-config the value yes as shown in Plate III

```
IPTABLES_MODULES_UNLOAD

IPTABLES_SAVE_ON_STOP

IPTABLES_SAVE_ON_RESTART
```

Plate III: IP Table config

Finally, the computer is instructed to forward IP packets, so that it can act as a router. To enable this, the following line is put in /etc/sysctl.conf:

net.ipv4.ip_forward=1

Then the system was restarted.

III. READING AND WRITING INTO A SUPERSIM

This section presents the process of programming a blank SuperSIM card, making use of python and other utility software like: PCSC and Pysim.

The software requirement for programming a SIM card includes;

i. Python 2.7.x or 3.6.x
ii. Pyscard
iii. Pysim
iv. Other dependencies
v. Linux server (fedora is used in this paper)

Note: windows server can also be used but will have a different configuration from that which will be described in the paper.

The hardware requirement includes:

i. A computer (laptop/desktop)
ii. Routers
iii. SuperSIM card (Universal SIM)
iv. SIM card reader.

The process is as shown in figure 3.

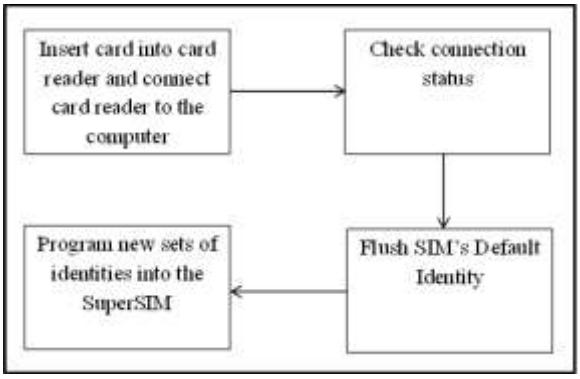

Figure 3: A block diagram showing how a blank SIM card is programmed

To read from a SIM card

The SIM card was inserted into the card reader and the card read was connected the configured computer having the appropriate requirement via the USB port. Then from a command shell on the Linux server, the status of the connection is checked by running

$ pcsc_scan

Some basic information about the Card reader and the SIM card is displayed to indicating that the connection was successful. Then this command *Ctrl+C* was executed to exit pcsc_scan. The information about the SIM card was read by running the following command

$ ./pySim-read.py –p0 or ./pySim-read.py –p1

How To Program or Write SuperSIM/USIM
The SIM card used was the 16-in-1 SuperSIM card, intended for COMP128v1 based cloning and allows to program up to 16 SIM identity in a single card [12]. New sets of SIM identities was added to the SIM by running the *./pySim-prog.py* command, specifying new sets required information. Like the Mobile Country Code (MCC), Mobile Network Code (MNC) KI (ki) etc.[2].The card is removed when the new set of profiles are programmed into the SIM safely.

IV. CONCLUSION

In this paper the step by step approach to configuring an ad-hoc network for wireless communication between profiles programmed into a blank SuperSIM card and a computer running a Linux server was presented. Python utility software was introduced and was used to program a blank SuperSIM card.

V. ACKNOWLEDGEMENT

This research work is sponsored by the Nigerian Communications Commission (NCC), Nigeria under research funding NCC/CS/007/15/C/040 of 2014.

REFERENCE

[1] Aibinu, A. M., Onumanyi, A. J., Adedigba, A. P., Ipinyomi. M., Folorunso, T. A., & Salami, M. J E. (2017). Development of hybrid artificial intelligent based handover decision algorithm. Engineering Science and Technology, an International Journal, 20(2), 381-390.

[2] Intelligence, G. S. M. A. (2015). Understanding SIM evolution.

[3] Salkintzis, A. K., Fors, C., &Pazhyannur, R. (2002). WLAN-GPRS integration for next-generation mobile data networks. IEEE Wireless communications, 9(5), 112-124.

[4] ISO 7816-3: Electronic Signals and Transmission Protocols of ISO7816 3

[5] Akkaladevi, S., Keesara, H., & Luo, X. (2011). Efficient forensic tools for handheld device: a comprehensive perspective. SoftwEng Res Manage Appl Stud ComputIntell, 377, 349-359.

[6] Jansen, W., & Ayers, R. (2006). Forensic Tools for Mobile Phone Subscriber Identity Modules. Journal of Digital Forensics, Security and Law, 1(2), 4.

[7] Mellars, B. (2004). Forensic examination of mobile phones. Digital Investigation, 1(4), 266-272.

[8] Jahankhani, H. (2009). Criminal investigation and forensic tools for smartphones. International Journal of Electronic Security and Digital Forensics, 2(4), 387-406.

[9] Ayers, R. P., Jansen, W., Delaitre, A. M., &Moenner, L. (2007). Cell phone forensic tools: an overview and analysis update (No. NIST Interagency/Internal Report (NISTIR)-7387).

[10] Anwar, N., Riadi, I., &Luthfi, A. (2016). Forensic SIM card cloning using authentication algorithm. International Journal of Electronics and Information Engineering, 4(2), 71-81.

[11] Han, J. U., Lee, Y. K., Jeon, C. M., Ryu, J., Hong, E. M., Yang, S., ... & Bang, H. (2009, May). Both nor and nand embedded hybrid flash for s-sim application using 90 nm process technology. In Memory Workshop, 2009. IMW'09. IEEE International (pp. 1-2). IEEE.