Timothy Kang
ITMS 555
Assign10

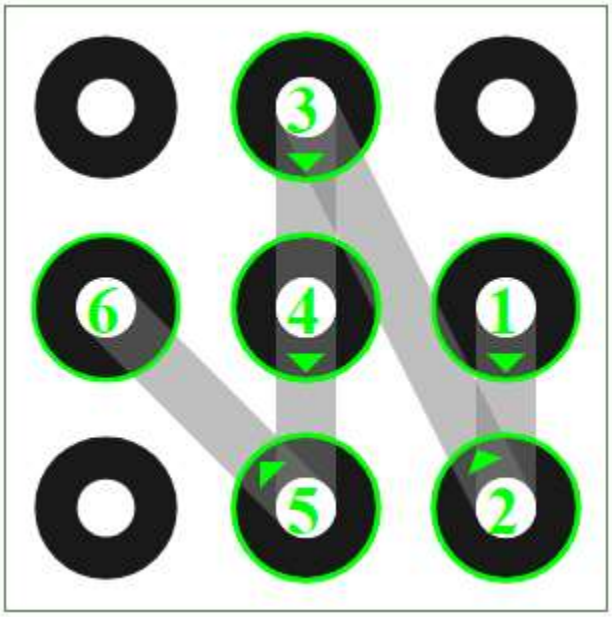

Fig 1.1: This is the pattern that is set on the phone (This corresponds to 5-8-1-4-7-3)

```
C:\WINDOWS\system32>adb devices
List of devices attached
04e05b709b4a1592        device
```

Fig 1.2: Making sure that phone is properly detected

```
C:\WINDOWS\system32>adb shell
shell@mako:/ $ su
root@mako:/ #
```

Fig 1.3: Starting up adb shell and gaining root privileges

```
root@mako:/ # cp /data/system/gesture.key  /sdcard/Download/
root@mako:/ # ls /sdcard/D
DCIM/     Download/
root@mako:/ # ls /sdcard/Download
SuperSU-v2.82.zip
gesture.key
locksettings.db
locksettings.db-wal
password.key
root@mako:/ #
```

Fig 1.4: Copying gesture key to sdcard/Download directory and confirming that it has been copied

```
D:\Users\tkang6\Desktop>adb pull -p /sdcard/Download/gesture.key .
/sdcard/Download/gesture.key: 1 file pulled. 0.0 MB/s (20 bytes in 0.180s)
```

Fig 1.5: Pulling gesture.key to desktop

```
D:\Users\tkang6\Desktop>python GenerateAndroidGestureRainbowTable.py
2018-11-19 18:39:43.846777: Building hashes for patterns with length 3
2018-11-19 18:39:43.856063: Building hashes for patterns with length 4
2018-11-19 18:39:43.911724: Building hashes for patterns with length 5
2018-11-19 18:39:44.283308: Building hashes for patterns with length 6
2018-11-19 18:39:47.834610: Building hashes for patterns with length 7
2018-11-19 18:40:13.253770: Building hashes for patterns with length 8
2018-11-19 18:42:31.882865: Building hashes for patterns with length 9

D:\Users\tkang6\Desktop>
```

Fig 1.6: Running script to create hash dictionary

```
D:\Users\tkang6\Desktop>python Android_GestureFinder.py gesture.key
Offset          Hash                                             Pattern
     -2012      9487570314d7d075fb59446693667970a1e5bf55         [5, 8, 1, 4, 7, 3]
```

Fig 1.7: Run another python script to lookup hash database and find pattern

Report:

The first figure shows the pattern that was set as a lock on my phone. The pattern length is 6 and if compared to a lock screen numbering, you can see that the pattern is 5,8,1,4,7,3. Unlike with PIN/password locks which require finding hash value in /data/system/password.key and salt value in /data/system/locksettings.db, hash values of pattern values are stored at /data/system/gesture.key. This was retrieved by copying the file onto the /sdcard/Downloads directory after rooting in. On a windows terminal, I can now pull the gesture.key file into any directory I want which in this case was my desktop. Since a pattern lock can only have a set number of possible patterns, I am able to do a dictionary attack. Simply put, this so-called dictionary will have every possible pattern so that the hash value of gesture.key can be compared with it and cracked. The GenerateAndroidGestureRainbowTable.py script was used to create the dictionary which is called AndroidLockScreenRainbow.sqlite. Another script called Android_GestureFinder.py can be run to solve the pattern as long as the gesture.key, python file,

and sqlite file are all in the same directory. Typically, the python file should be edited to work properly for cracking but fortunately for me, it was already done on RADISHng. Once you run this script, a hash is displayed along with the pattern. The pattern it outputted was indeed the correct pattern lock on my phone.