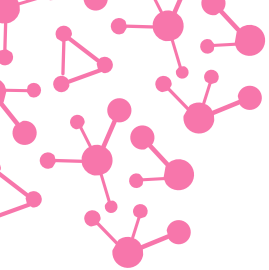
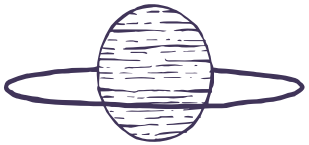
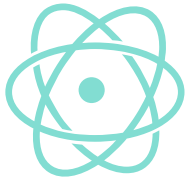






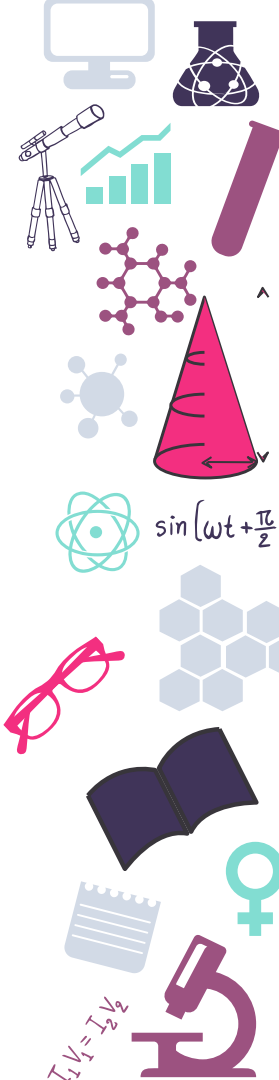
Android Unlock Pattern Security

Team 4

Team members:

Tong Wu (tw445)
Xun Tang (xt63)
XInyu Lyu (xl422)

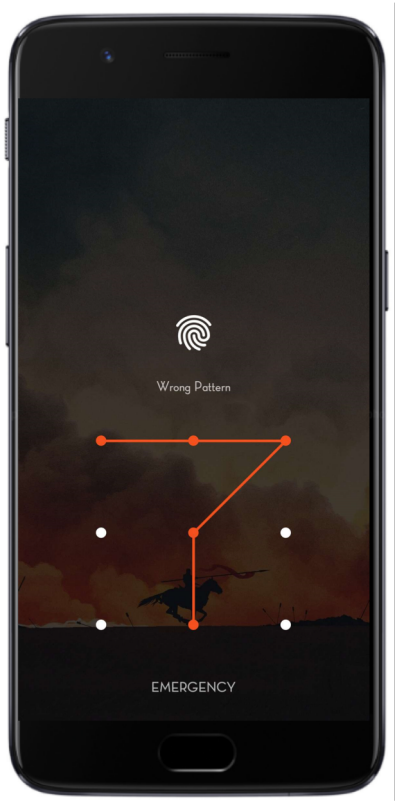
You can contact us at
tw445@scarletmail.rutgers.edu
xt63@scarletmail.rutgers.edu
xl422@scarletmail.rutgers.edu



Motivations:

Android unlock pattern is one of the most common unlock methods in our daily life, so we want to know:
How secure exactly are these patterns?

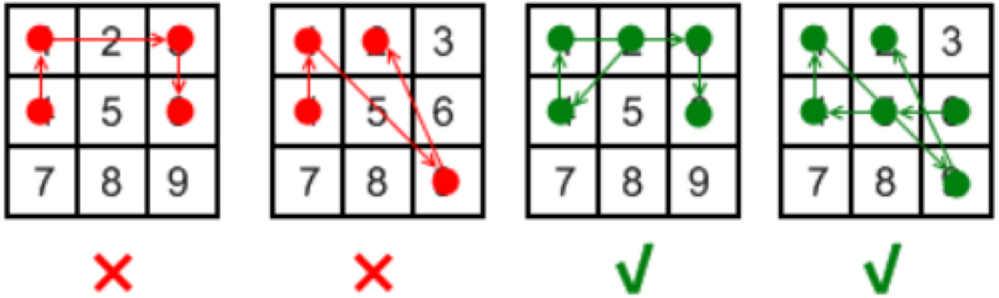
Security ≈ Number of Valid Gestures



Valid Unlock Gesture:

Rules:

- Gesture must be consecutive, keys must be distinct.
- No jump through unselected key.
- The order of keys used matters.



Algorithms

Brute Forth method

DFS method

BackTracking method

DP method

A Smarter Way

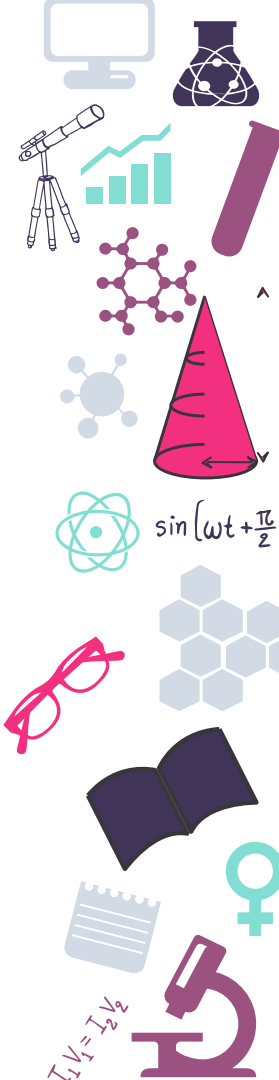
1 Brute Force

Permutation

Use recursion method to get all permutations of possible gestures.

Find valid ones

According to invalid permutations stored in hashmap, find the number of valid gestures from previous part.



BackTracking

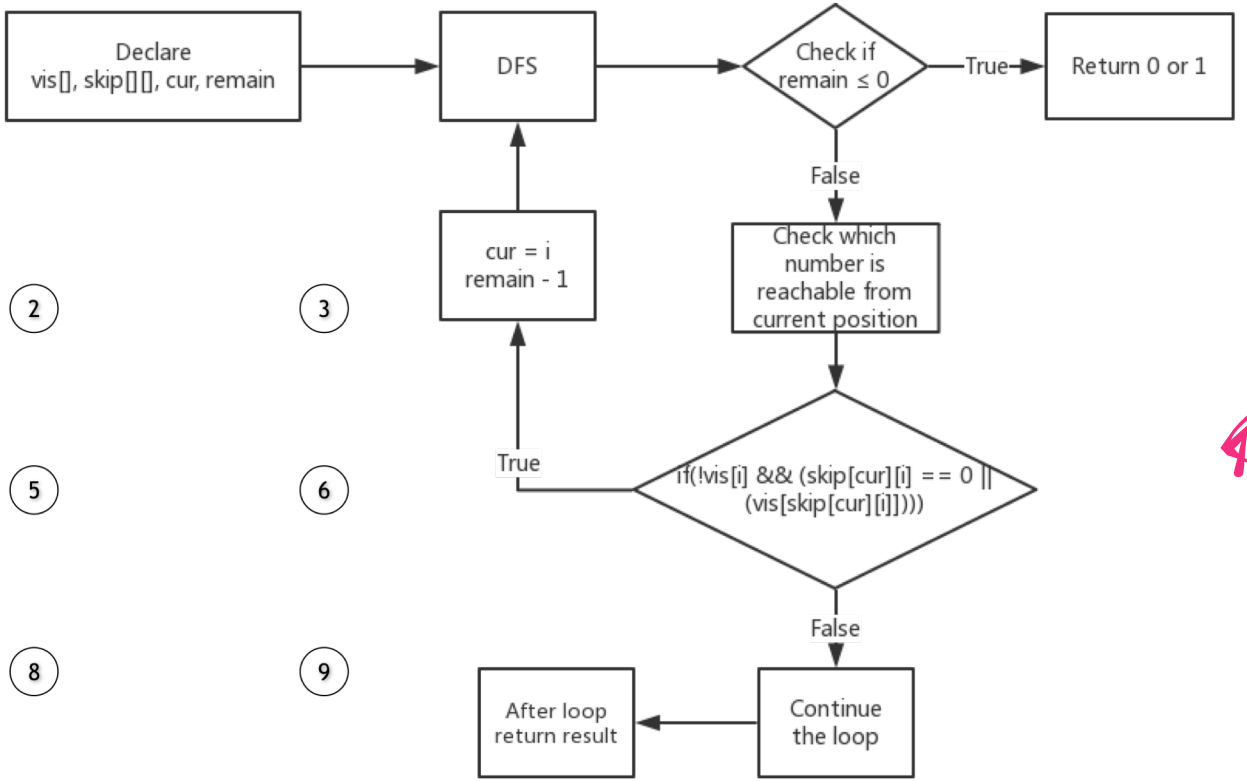
2

- Continue selecting an unused vertex until the length is satisfied.
- Check if it is valid before adding it to gestures.
- If it is not valid, the algorithm rejects the current vertex, backtracks and continues.

demo

3 DFS Implementation

Basic Idea



1 2 3

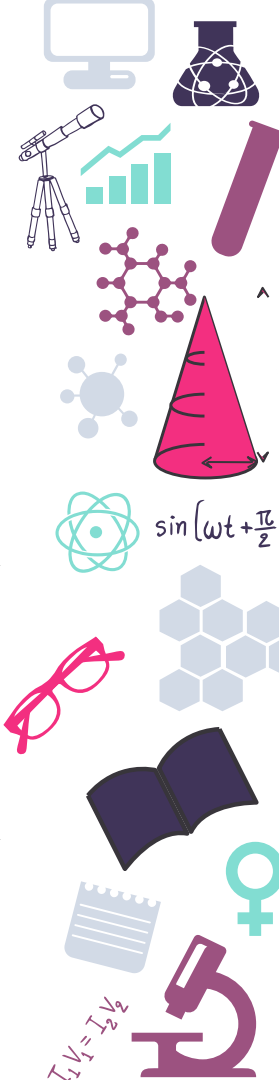
4 5 6

7 8 9

Optimization

Because of symmetry, 1, 3, 7, 9 and 2, 4, 6, 8 can be calculated in the same way.

```
for(int i = m; i <= n; ++i) {
    rst += DFS(vis, skip, 1, i - 1) * 4;    % 1, 3, 7, 9 are symmetric
    rst += DFS(vis, skip, 2, i - 1) * 4;    % 2, 4, 6, 8 are symmetric
    rst += DFS(vis, skip, 5, i - 1);        % 5
}
```



4 A Smarter Way

isVisited :

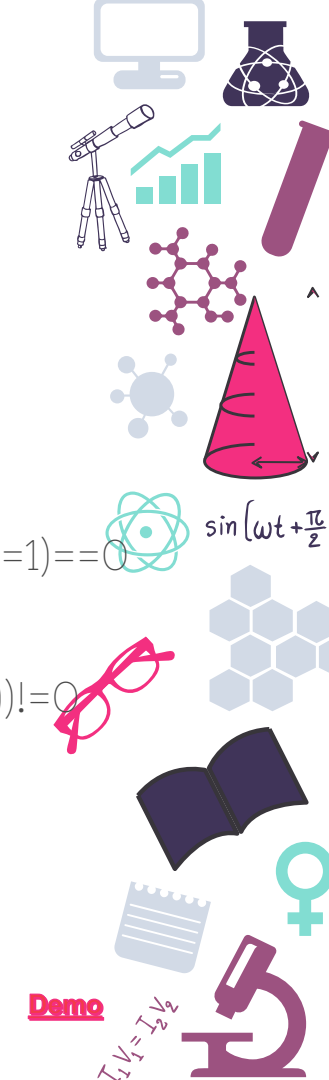
- Update “used” (9-bit bitmask) :
  used2 = used | (1 << (i * 3 + j))
- Judge if visited :
  used2 > used

- Space complexity is O(1)

Graphical relationship

- Find middle node :
  ((i1+i2) % 2) ==1 || (j1+j2) % 2)==1)==0
- If exists middle node :
  used & (1 << (I / 2 * 3 + J / 2))!=0

- Time complexity is O(9^n)



Dynamic Programming 5

dp[len][state][endNode]

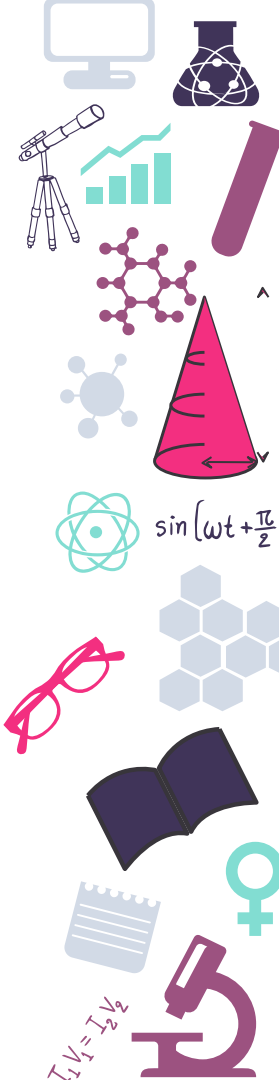
- Android pattern table
- Pattern length [len]
- End as [endNode]
- Visited nodes [state] (9-bit bitmask)
- nextState = (state | (1 << (i – 1)));

void calcDP(n)

- Set up start node :
  “dp[1][1 << (i – 1)][i] = 1”
- Traverse, find pattern with previous node :
  “dp[len][state][endNode]!=0”
- Find next valid node :
  “skip[endNode][i] == 0 || ((state & (1 << (skip[endNode][i] – 1))) != 0)”
- Update pattern table

Experiments

Intel Core i5-5257U with 8GB memory and 64 bit build Late 2015 MacBook Pro. The inversion of Java is 1.8.0_144-b01.

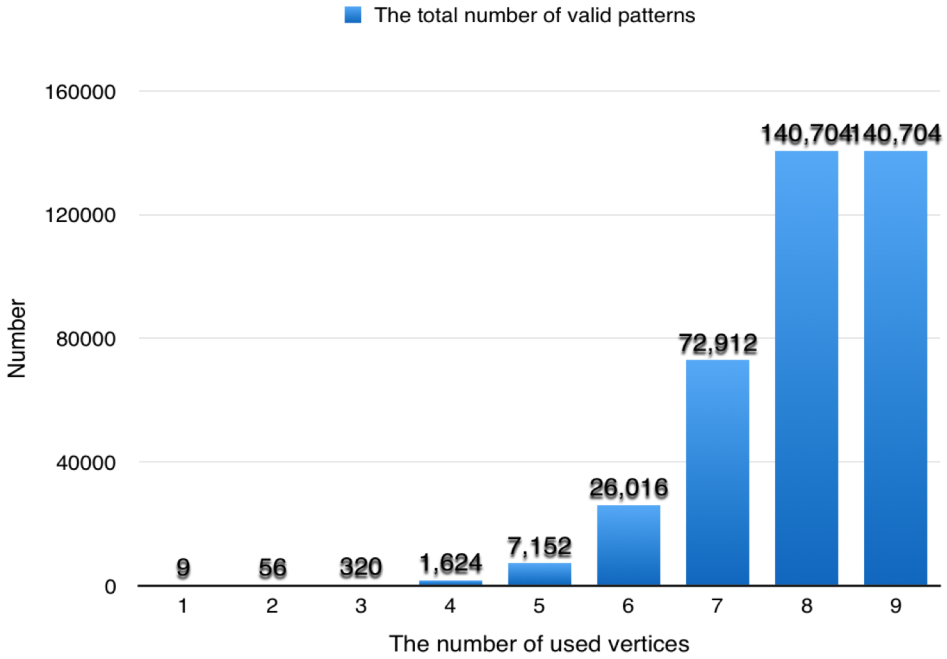


Results and Performance

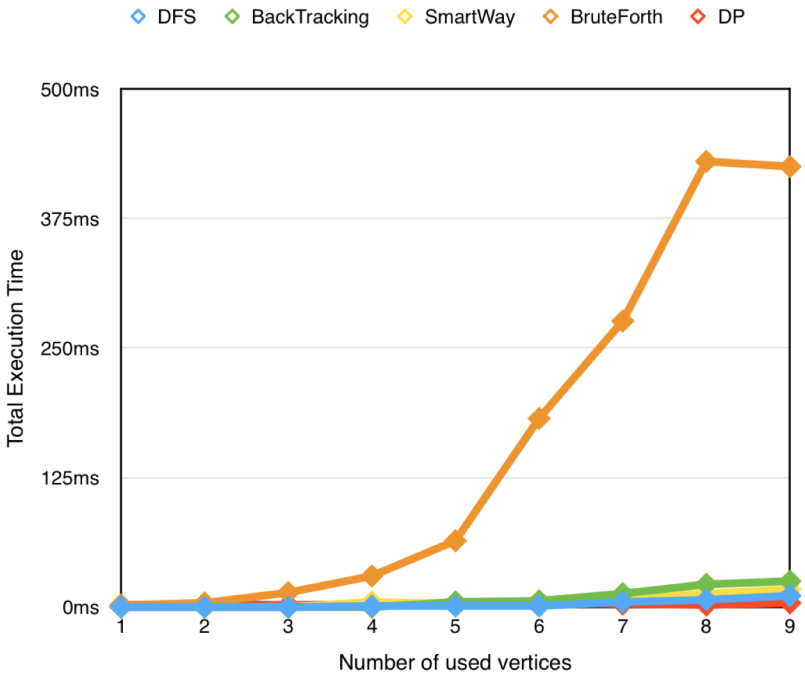
Result

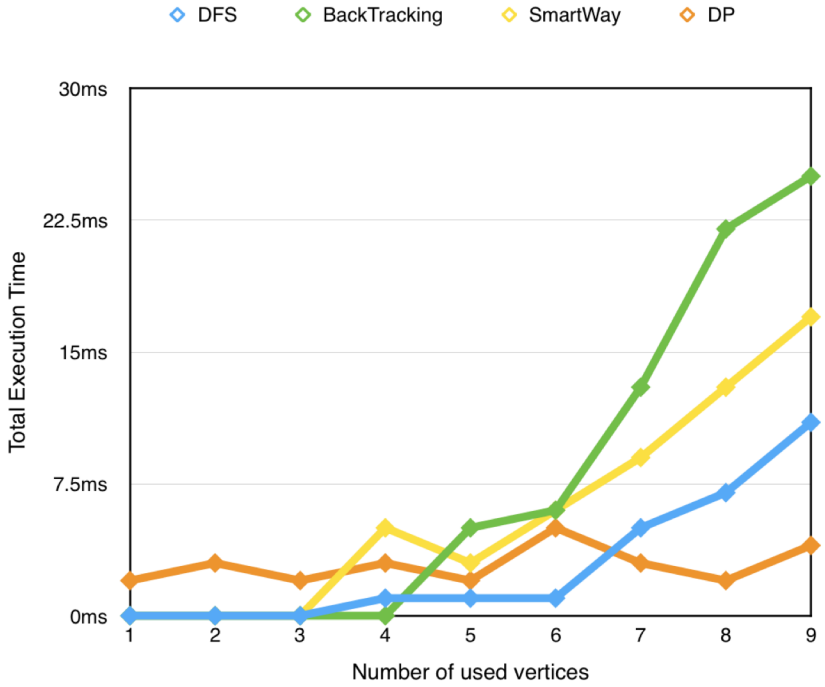
Security of gestures with 8, 9 vertices ≈ 5 digits PIN.

However, most people use 5 to 7 vertices, ≈ iPhone 4 digits PIN.



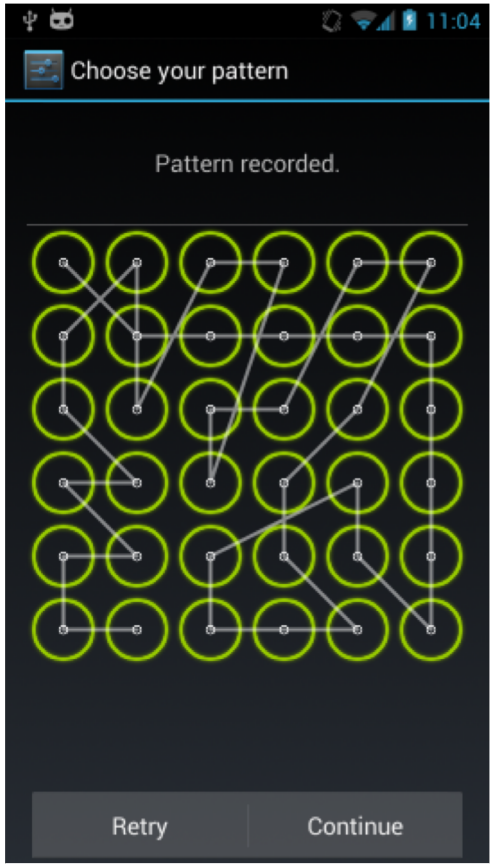
Performance





Future Work

In the future, we'd like to expand the size of pattern to 4*4, 5*5 and even more complicated.



Thanks!

Any questions?

You can find us at:
tw445@scarletmail.rutgers.edu
xt63@scarletmail.rutgers.edu
xl422@scarletmail.rutgers.edu