

I Know Where You Are: 基于LBSN APP的地理位置追踪

赵 双(DFlower)
羅夏樸(Xiapu Luo)

2015年9月1日

Who AM I

2

- 赵双 (DFlower, dflower.zs@gmail.com)
 - Insight-Labs Team 成员 (<http://insight-labs.org>)
 - 研究领域：漏洞挖掘 / 恶意代码分析 / 手机安全
 - 《0day安全：软件漏洞分析技术》(第2版)作者之一
 - OWASP China 2010、 Xcon 2011 Speaker

- 羅夏樸 (Xiapu Luo, luoxiapu@gmail.com)
 - RAP in HKPolyU
 - 研究領域：手機安全，網絡安全
 - Papers published in major security conferences (e.g., NDSS, BlackHat, Defcon, etc.)

□ LBSN APP 介绍

□ 基于LBSN APP进行地理位置追踪

□ 实例: 追踪全北京新浪微博用户的地理位置

□ 给APP厂商的建议

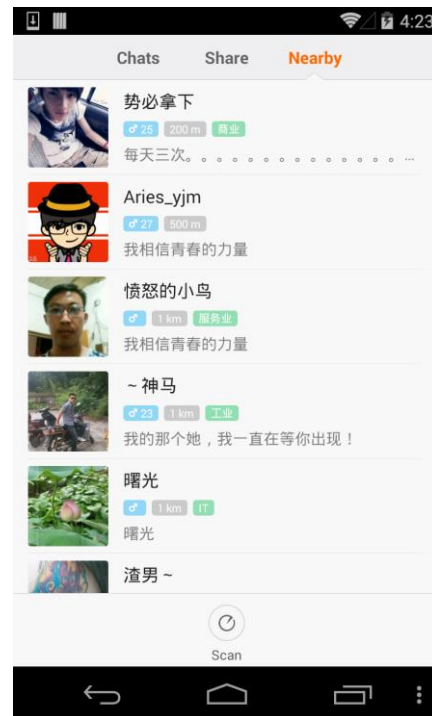
LBSN APPs

4

- LBSN: Location-based Social Network, 基于位置的社交网络
- 很多LBSN APP具有“发现附近的人(Nearby)”功能



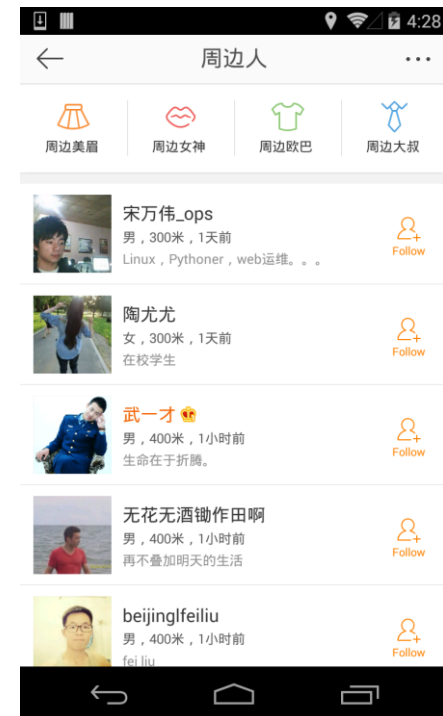
微信 Wechat



米聊 Mitalk



陌陌 Momo

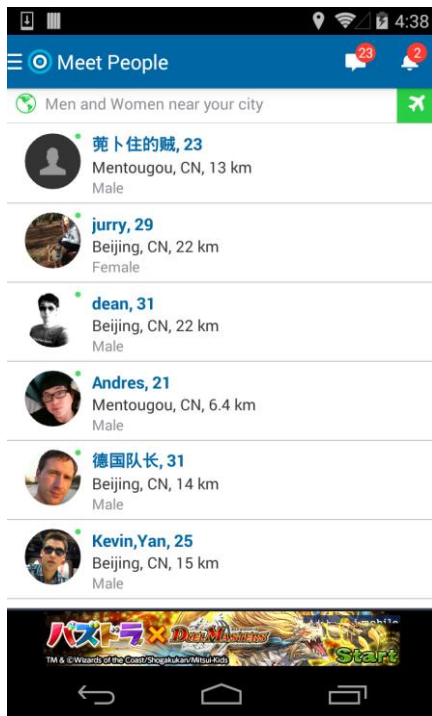


新浪微博 Weibo

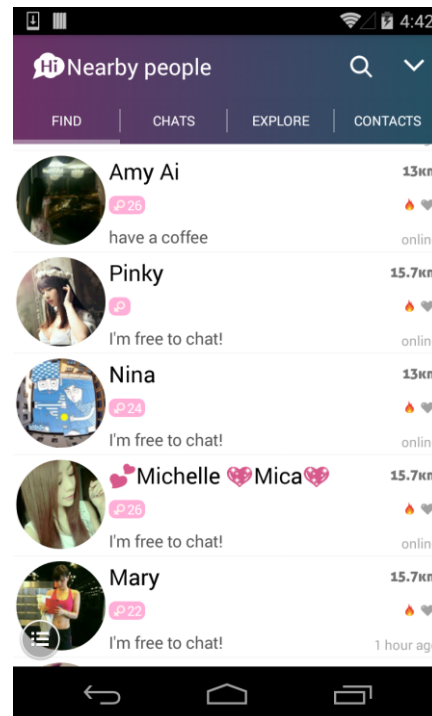
LBSN APPs

5

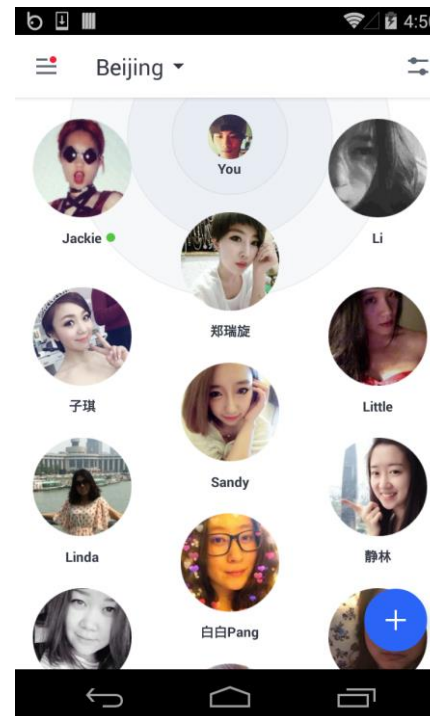
➤ 很多LBSN APP具有“发现附近的人(Nearby)”功能



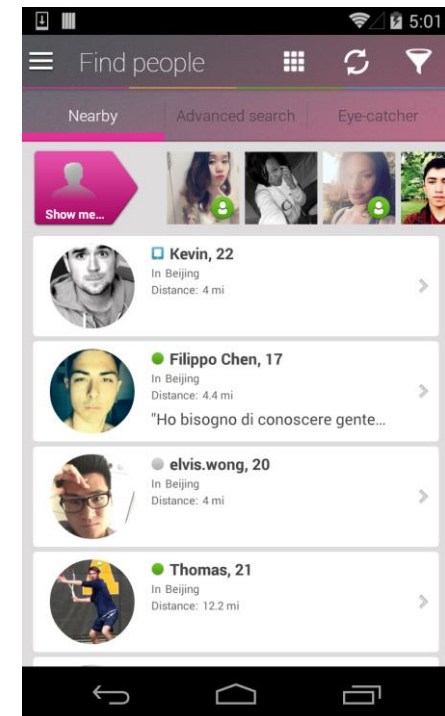
 SKOUT



 SayHi



 Badoo



 LOVOO

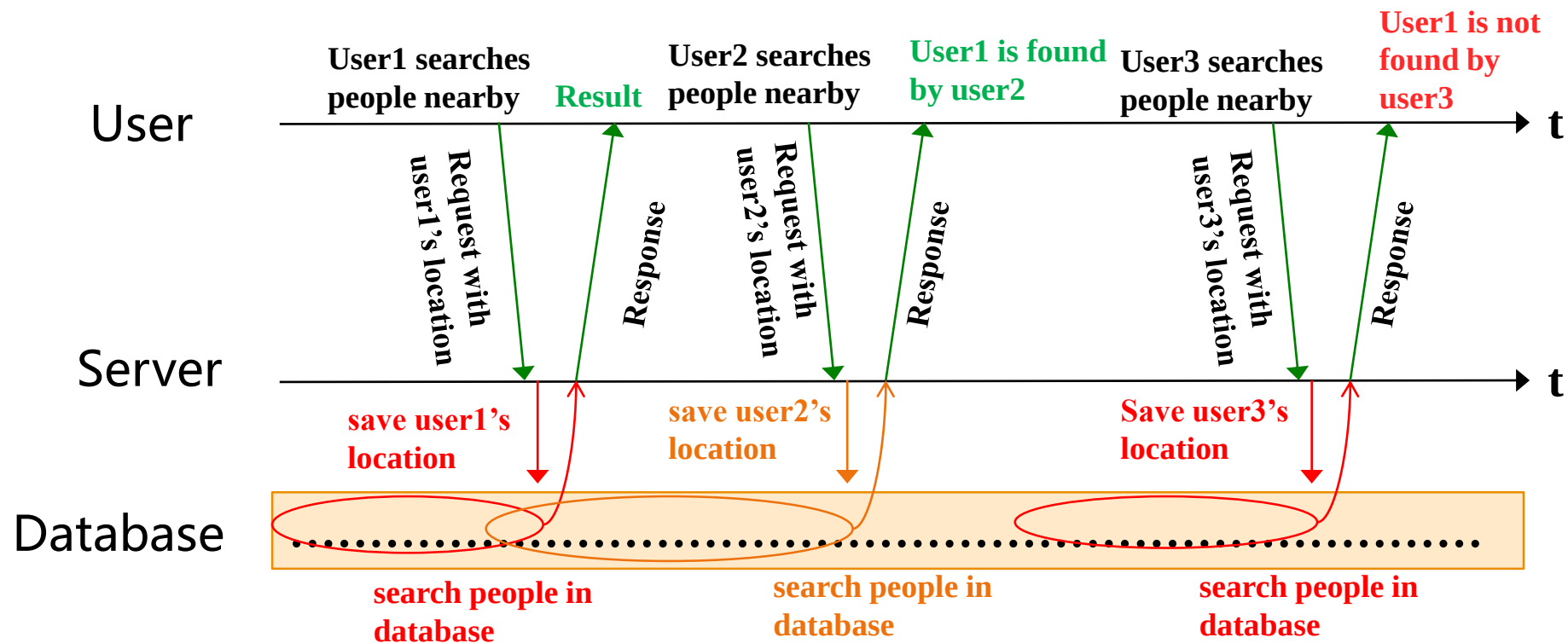
LBSN APPs

6

- 很多LBSN APP具有“发现附近的人(Nearby)”功能

当用户使用Nearby功能时：

- 上传自己位置信息
- 寻找自己所处位置周边的陌生人
- 允许其他人在**一定时间内**发现自己



✓ LBSN APP 介绍

□ 基于LBSN APP进行地理位置追踪

- 数据发送
- 数据获取
- 数据挖掘
- DEMO: Mitalk, Wechat, Weibo

□ 实例: 追踪全北京新浪微博用户的地理位置

□ 给APP厂商的建议

如何基于LBSN APP进行位置追踪？

1. 数据发送：

登录APP并在不同的地理位置搜索附近的人

2. 数据获取：

获得 { 时间，地点，人 } 数据集

3. 数据挖掘：

数据集信息足够多的情况下，可对目标的日常活动范围和轨迹进行追踪

针对不同类型的APP，采用不同的数据发送方式：

1. 构造数据包
2. 模拟器仿真

APP信息			数据安全性		利用难度	
APP	Googleplay下载量 (million)	360电子市场下载量 (million)	位置精确度(APP中)	数据传输方式	爬取APP位置数据方式	利用难度
Weibo	5-10	456	100m	明文	构造数据包	低
MeetMe	10-50	0.001	100m	明文	构造数据包	低
Skout	10-50	0.06	1000m	明文	构造数据包	低
SayHi	10-50	0.04	100m	明文	构造数据包	低
Badoo	50-100	0.07	1000m	SSL单向认证 (参数不明)	构造数据包 / 模拟器仿真	?
Momo	1-5	168	10m	SSL单向认证	构造数据包	低
Mitalk	0.5-1	17	100m	带加密参数的明文	破解加密参数&构造数据包 / 模拟器仿真	高
LOVOO	10-50	0.001	0.1mile	SSL单向认证，带 加密参数	破解加密参数&构造数据包 / 模拟器仿真	高
Wechat	100-500	455	100m	SSL双向认证	替换证书&构造数据包 / 模拟器仿真	高

1. 数据发送

10

1. APP数据明文传输

典型APP：Weibo，Meetme，SayHi，Skout

方式：构造并发送包含特定地理位置的数据包，获取该位置附近的人

- Weibo

GET

http://api.weibo.cn/[REDACTED]?gender=0&sourcetype=findfriend&offset=0&
[REDACTED]c=android&lat=39.83178&long=116.290966&page=1&sort=1&count=20↵

- Meetme

GET ↵

http://friends.meetme.com/[REDACTED]?placement=meet&latitude=38.988088&longitude=76.977333&targetGender=b&orderB
y=distance&includeFriends=t&onlineOnly=f&pageSize=30↵

- SayHi

GET http://r.x-vv.com/[REDACTED]>=true&ii=1&ts=0&of=0&lc=116.2085999,39.9726842↵

- Skout

POST http://and.skout.com/api/1/me/location 设置 cityname↵

GET

http://and.skout.com/[REDACTED]?application_code=[REDACTED]&area=city&limit=20&start=0&rand_
token=[REDACTED]↵

1. 数据发送

11

2. APP数据采用HTTPS传输（SSL单向认证）

典型APP：Momo 2014

方式：使用伪造证书的方式还原HTTPS数据包，然后构造并发送包含特定地理位置的数据包，获取该位置附近的人

- Momo

POST https://[REDACTED].immomo.com+

Count=20&lat=39.83178&lng=116.290966&index=0

1. 数据发送

12

3. APP数据采用HTTPS传输 (SSL双向认证/SSL Pinning)

典型APP : Wechat, Momo 2015

方式 : Android模拟器 + UIAutomator等自动化测试架构模拟用户操作 , 通过修改模拟器地理位置获取获取该位置附近的人

模拟Wechat用户操作 :

```
getUiDevice().pressHome();  
UiObject wxApp = new UiObject(new UiSelector().text( "WeChat" ));  
wxApp.clickAndWaitForNewWindow();  
UiObject discoverTab = new UiObject(new UiSelector().text( "Discover" ));  
    "Discover" 标签  
UiSelector uiSelector = new UiSelector().text( "People Nearby" );  
    Nearby "按钮"  
UiObject nearbyButton = new UiObject(uiSelector);  
nearbyButton.clickAndWaitForNewWindow();  
    Nearby "按钮"
```

```
//点击 "Home"  
    //寻找Wechat图标  
//启动Wechat  
    //点击  
    //寻找 "People"  
  
//点击 "People"
```

1. 数据发送

13

4. APP数据明文传输，但包含加密参数

典型APP：Mitalk，LOVOO

抓取方式：

1) 破解加密参数，构造并发送包含特定地理位置的数据包，获取该位置附近的人

2) Android模拟器+自动化测试框架

• 数据包分析

1) 抓取APP获取“附近的人”时发送的数据包

2) 更改数据包中的latitude/longitude值后重放

3) 返回Error

4) 多次对比APP发送的数据包发现，uuid是账号，token在登录之后保持不变，参数s是对所有其他参数的校验，服务器通过校验s的值来判断数据包是否被篡改

The screenshot shows a Wireshark packet capture of a GET request. The request URL is `GET /find/v3/user/823111239/nearby2/user?count=20&latitude=39.856599&longitude=116.287235&setting=%7B%22stranger%22%3Atrue%2C%22sex%22%3A%22M%22%7D&start=0&time=1435583249793&uuid=823111239&s=6CE47C07FF0ECFDE43680790257C455A&token=ODIzMTBxMjM5OjEOMzUzNzYONDI6YS0xMDBfMV9hbmRyb2lkLTBxNjoteDE%3D`. The packet is selected in the packet list, and the packet details pane shows the raw data. The raw data shows the HTTP response: `HTTP/1.1 200 OK`, `Server: Tengine/2.1.0`, `Date: Mon, 29 Jun 2015 13:07:29 GMT`, `Content-Type: application/json; charset=utf-8`, `Connection: keep-alive`, `Content-Length: 4096`. The raw data also shows the JSON response: `{"result": "ok", "description": "成功", "data": {"total": 20, "next": 20, "list": [{"i`.

The screenshot shows a Wireshark packet capture of a GET request. The request URL is `GET /find/v3/user/823111239/nearby2/user?count=20&latitude=38.856599&longitude=116.287235&setting=%7B%22stranger%22%3Atrue%2C%22sex%22%3A%22M%22%7D&start=0&time=1435583249793&uuid=823111239&s=6CE47C07FF0ECFDE43680790257C455A&token=ODIzMTBxMjM5OjEOMzUzNzYONDI6YS0xMDBfMV9hbmRyb2lkLTBxNjoteDE%3D`. The packet is selected in the packet list, and the packet details pane shows the raw data. The raw data shows the HTTP response: `HTTP/1.1 401 Unauthorized`, `Server: Tengine/2.1.0`, `Date: Mon, 29 Jun 2015 13:07:29 GMT`, `Content-Type: application/json; charset=utf-8`, `Connection: keep-alive`, `Content-Length: 4096`. The raw data also shows the JSON response: `{"result": "error", "description": "参数s错误", "data": {"total": 0, "next": 0, "list": []}}`.

DEMO - mitalk加密参数破解

15

- 破解参数s的计算过程

逆向mitalk的APK (apktool/dex2jar/JD-GUI.....) ,寻找s参数的生成过程

```
public static boolean a(Context paramContext, String paramString1, String paramString2, List<NameValuePair> paramList)
{
    if ((TextUtils.isEmpty(paramString1)) || (TextUtils.isEmpty(paramString2))) {
        return false;
    }
    paramList.add(new BasicNameValuePair("s", URLEncoder.encode(a(paramList, paramString1))));
    paramList.add(new BasicNameValuePair("token", paramString2));
    return true;
}
```

s由paramList和paramString1计算而来，跟踪a(paramList, paramString)函数

```
public static String a(List<NameValuePair> paramList, String paramString)
{
    paramList.add(new BasicNameValuePair("time", String.valueOf(System.currentTimeMillis())));
    Collections.sort(paramList, new bb());
    StringBuilder localStringBuilder = new StringBuilder();
    Iterator localIterator = paramList.iterator();
    for (int m = 1; localIterator.hasNext(); m = 0)
    {
        NameValuePair localNameValuePair = (NameValuePair)localIterator.next();
        if (m == 0) {
            localStringBuilder.append("&");
        }
        localStringBuilder.append(localNameValuePair.getName()).append("=").append(localNameValuePair.getValue());
    }
    localStringBuilder.append("&").append(paramString);
    return com.xiaomi.channel.d.f.d.c(new String(com.xiaomi.channel.d.f.a.b localStringBuilder.toString().getBytes()));
}
```

s = b(name1=value1&name2=value2&...¶mString1)

可以看出，函数把paramList中的各个参数按照字母顺序排序后用

“[name1]=[value1]&[name2]=[value2]...” 的形式连接，最后再加上"¶mString1"

DEMO - mitalk加密参数破解

16

- 破解参数s的计算过程

将apk反编译为smali代码，加入Logcat代码打印paramString1，之后重打包APK，安装到手机，运行mitalk并查看Logcat日志，即可得到paramString1的值（也是登录之后固定不变的）

	Tag	Text
2796 :cond 1		
2797 const-string/jumbo v1, "DFLOWER"	DFLOWER	9B90FAAF417282AC8C5267EF4DC968B7
2798	DFLOWER	F7CE178FA1BD1A3924A52B77F4C9A084
2799 invoke-static {v1, p1}, Landroid/util/Log;->d(Ljava/lang/String;Ljava/lang/String	DFLOWER	count=100&read_time=1411458647011&tir
2800		uuid=539405118&F7CE178FA1BD1A3924A52
2801 invoke-static {p3, p1}, Lcom/xiaomi/channel/common/network/az;->a(Ljava/util/List	DFLOWER	3B751743D61D0315CA007B9107578C2C
2802	DFLOWER	F7CE178FA1BD1A3924A52B77F4C9A084
2803 move-result-object v0	DFLOWER	cmxsts=-1&time=1411459705330&uuid=53
2804		7F4C9A084
2805 const-string/jumbo v2, "DFLOWER"	DFLOWER	21530623E7057C3A23C0922CC86B3B50
2806	DFLOWER	F7CE178FA1BD1A3924A52B77F4C9A084
2807 invoke-static {v2, v0}, Landroid/util/Log;->d(Ljava/lang/String;Ljava/lang/String	DFLOWER	count=100&read_time=1411458757433&tir
2808		uuid=539405118&F7CE178FA1BD1A3924A52
2809 new-instance v1, Lorg/apache/http/message/BasicNameValuePair;		1AA1B43607746DC3FCBC1BEB720751CC
2810	DFLOWER	
2811 const-string/jumbo v2, "s"		

至此，我们得到的结果是：

s = b(name1=value1&name2=value2&...¶mString1)，每个参数的值都是已知的，只要破解出加密算法com.xiaomi.channel.d.f.a.b就可以了

DEMO - mitalk加密参数破解

17

- 破解参数s的计算过程

而函数com.xiaomi.channel.d.f.a.b也是用Java写的，因此也很容易被反编译

(比如使用dex2jar/JD-GUI/
AndroChef Java Decompiler等)

至此，s参数破解完成，便可以用
程序来模拟mitalk在不同的地理
位置发送数据包获得“附近的
人”了。

```
public static char [] b = new char[64];
//public static byte[] c;
public static void Init()
{
    char var0 = (char)65;

    int var1;
    int var16;
    for (var1 = 0; var0 <= 90; var1 =
    {
        char[] var15 = b;
        var16 = var1 + 1;
        var15[var1] = var0++;
    }

    int var14;
    for (char var2 = (char)97; var2 <
    {
        char[] var13 = b;
        var14 = var1 + 1;
        var13[var1] = var2++;
    }
}
```

```
public static char[] a(byte[] var0, int var1, int var2)
{
    int var3 = (2 + var2 * 4) / 3;
    char[] var4 = new char[4 * ((var2 + 2) / 3)];
    int var5 = var1 + var2;

    int var10;
    for (int var6 = 0; var1 < var5; var1 = var10)
    {
        int var7 = var1 + 1;
        int var8 = 255 & var0[var1];
        int var9;
        if (var7 < var5)
        {
            int var21 = var7 + 1;
            var9 = 255 & var0[var7];
            var7 = var21;
        }
        else
        {
            var9 = 0;
        }
    }
}
```

```
int var12;
for (char var3 = (char)48; var3 <
{
    char[] var11 = b;
    var12 = var1 + 1;
    var11[var1] = var3++;
}
}
```

```
int var11;
if (var7 < var5)
{
    var10 = var7 + 1;
    var11 = 255 & var0[var7];
}
else
{
    var11 = 0;
}
```

```
name:铃铛, sex:男, distance:3318, desp:自由的水边
Target Found around 自由的水边!
name:铃铛, sex:女, distance:3260, desp:none
Target Found around 自由的水边!
name:上来自己动, sex:女, distance:3340, desp:去一7看看再说了没有
2014/9/26 9:18:18, Probing 自由的水边 ...:Find 113 Persons
2014/9/26 9:18:21, Probing 自由的水边 ...:Find 117 Persons
2014/9/26 9:18:24, Probing 自由的水边 ...:Find 121 Persons
2014/9/26 9:18:28, Probing 自由的水边 ...:Find 114 Persons
2014/9/26 9:18:31, Probing 自由的水边 ...:Find 107 Persons
2014/9/26 9:18:50, Probing 自由的水边 ...:Find 112 Persons
```

2. 数据接收

18

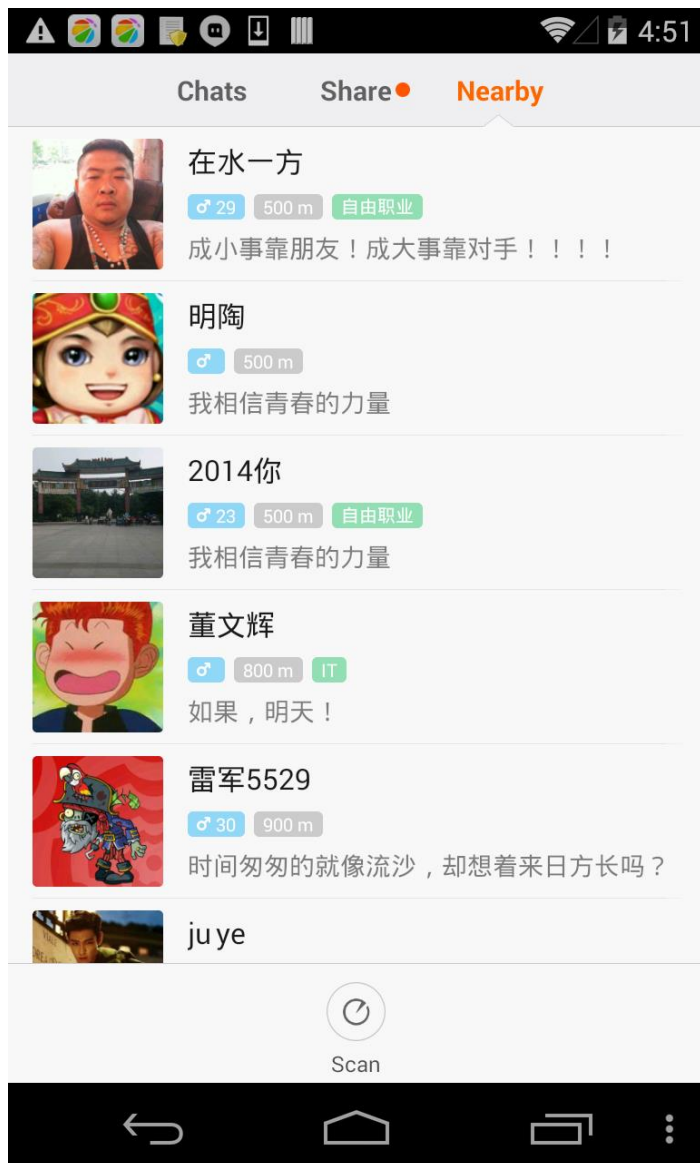
1. 构造数据包方式：
返回结果解析(Json/XML...)

2. 模拟器仿真方式：
UIAutomator识别界面文字

2. 数据接收

19

- 返回结果解析
米聊 (Mitalk)



2. 数据接收

20

- 返回结果解析

很多APP虽然在界面中降低了地理位置精度（如只精确到100m），但是在抓取到的数据包中，却能获取到许多额外的结果

2. 数据接收

21

- 返回结果解析

陌陌 (Momo)

APP中地理位置精度：0.01km=10m (距离)

数据包中地理位置精度：0.00000000000001m (距离)



2. 数据接收

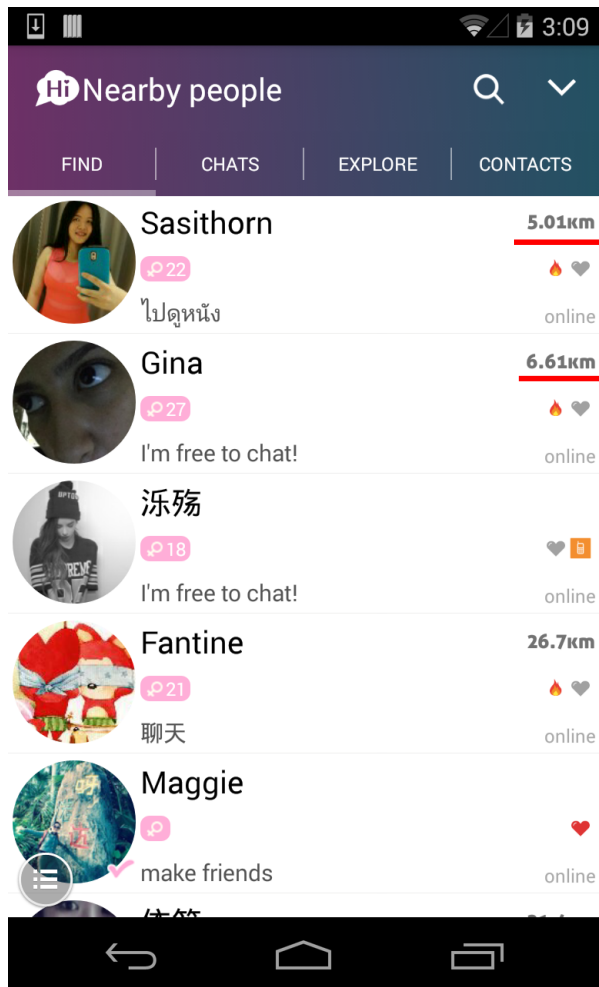
22

- 返回结果解析

SayHi

APP中地理位置精度：0.01km=10m（距离）

数据包中地理位置精度：经纬度 $0.000001^\circ \approx 0.1\text{m}$ （坐标）



```
JSON
{
  "d": {
    "g": 1,
    "h": 20336151,
    "img": "hi-icon_20336151_1433928063260",
    "j": 34359738374,
    "lc": {
      "lat": 116.339193,
      "lon": 39.9923481
    },
    "ls": 1435388151144,
    "n": "Sasithorn",
    "nn": 14749270016,
    "p": 250,
    "sm": "0,18,25,ไปดูหนัง",
    "ut": 0,
    "v": 6
  },
  "g": 1,
  "h": 20642901,
  "img": "hi-icon_20642901_1435323083398",
  "j": 34359738374,
  "lc": {
    "lat": 116.3573074,
    "lon": 39.9869576
  },
  "ls": 1435387803422,
  "n": "Gina",
  "nn": 13063159808,
  "p": 150,
  "ut": 0,
  "v": 5
}
```

2. 数据接收

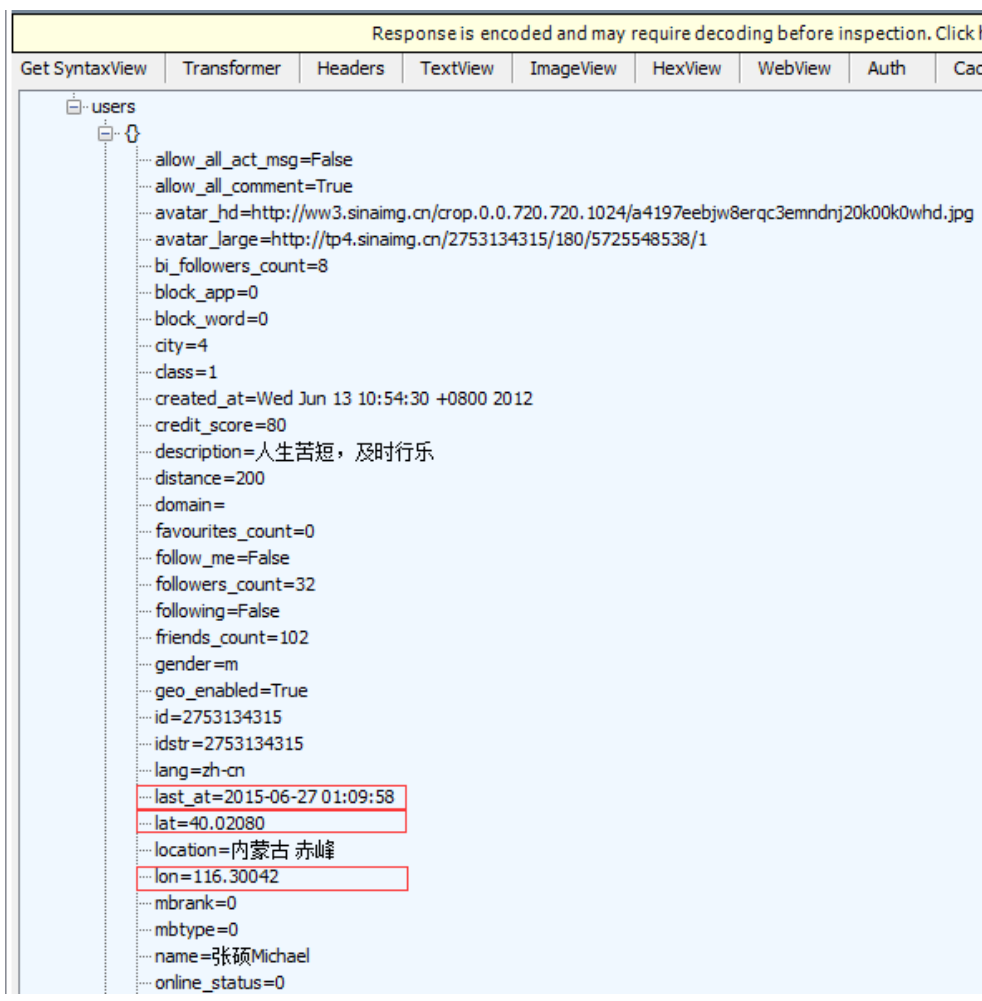
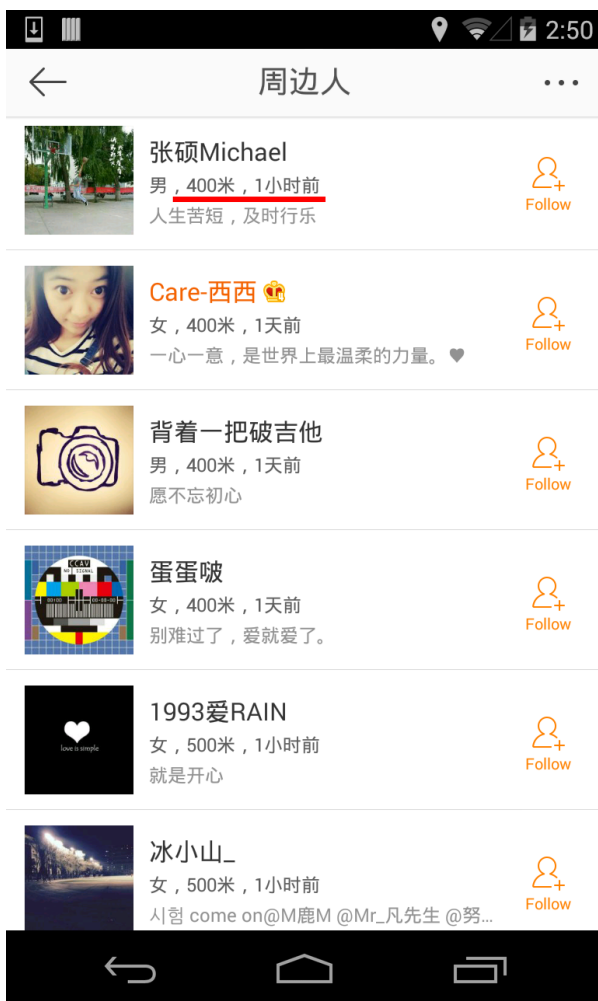
23

- 返回结果解析

新浪微博 (Weibo)

APP中地理位置精度：100m (距离)

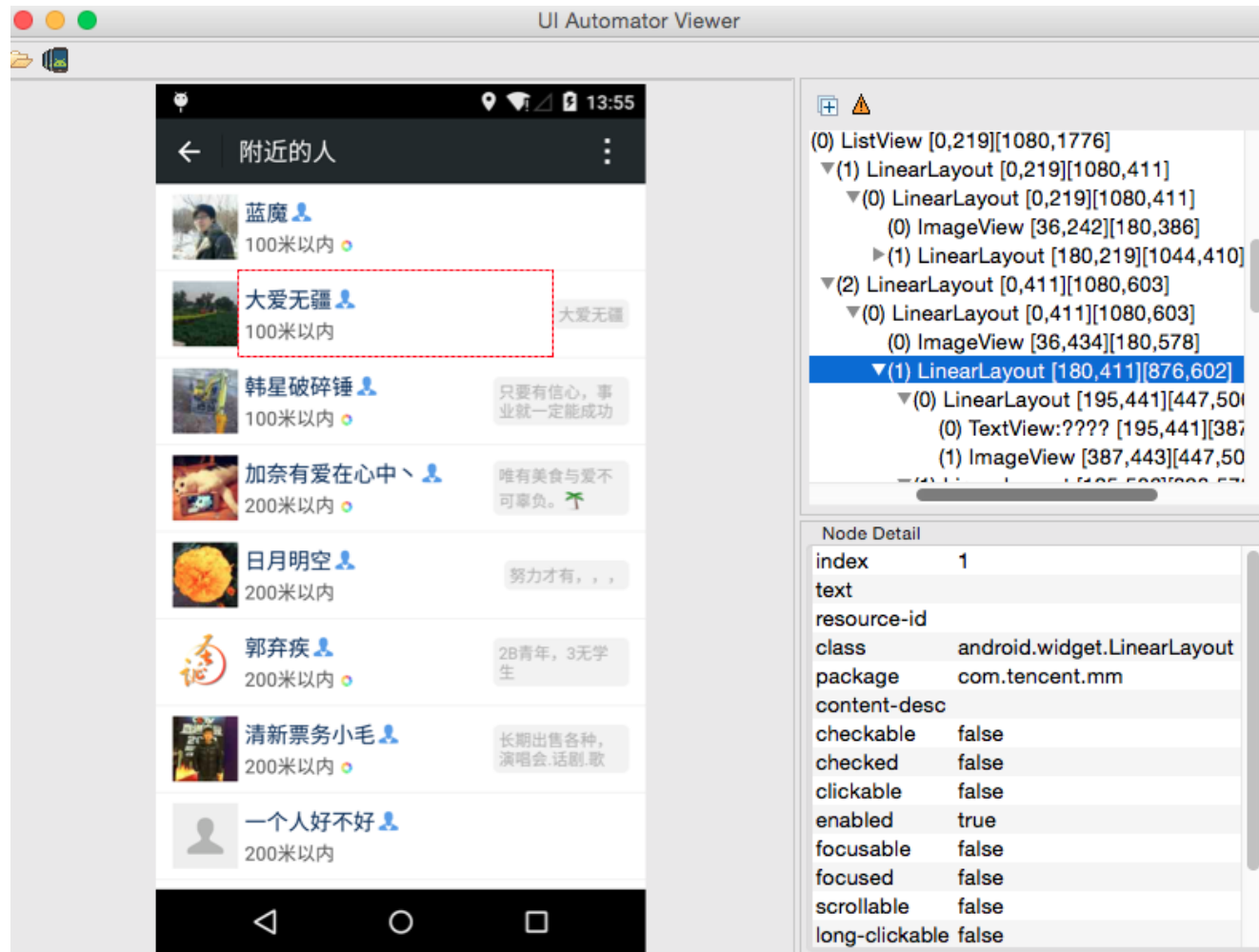
数据包中地理位置精度：经纬度 $0.00001^{\circ} \approx 1\text{m}$ (坐标) & 上次出现时间



2. 数据接收

- UIAutomator识别
微信 (Wechat)

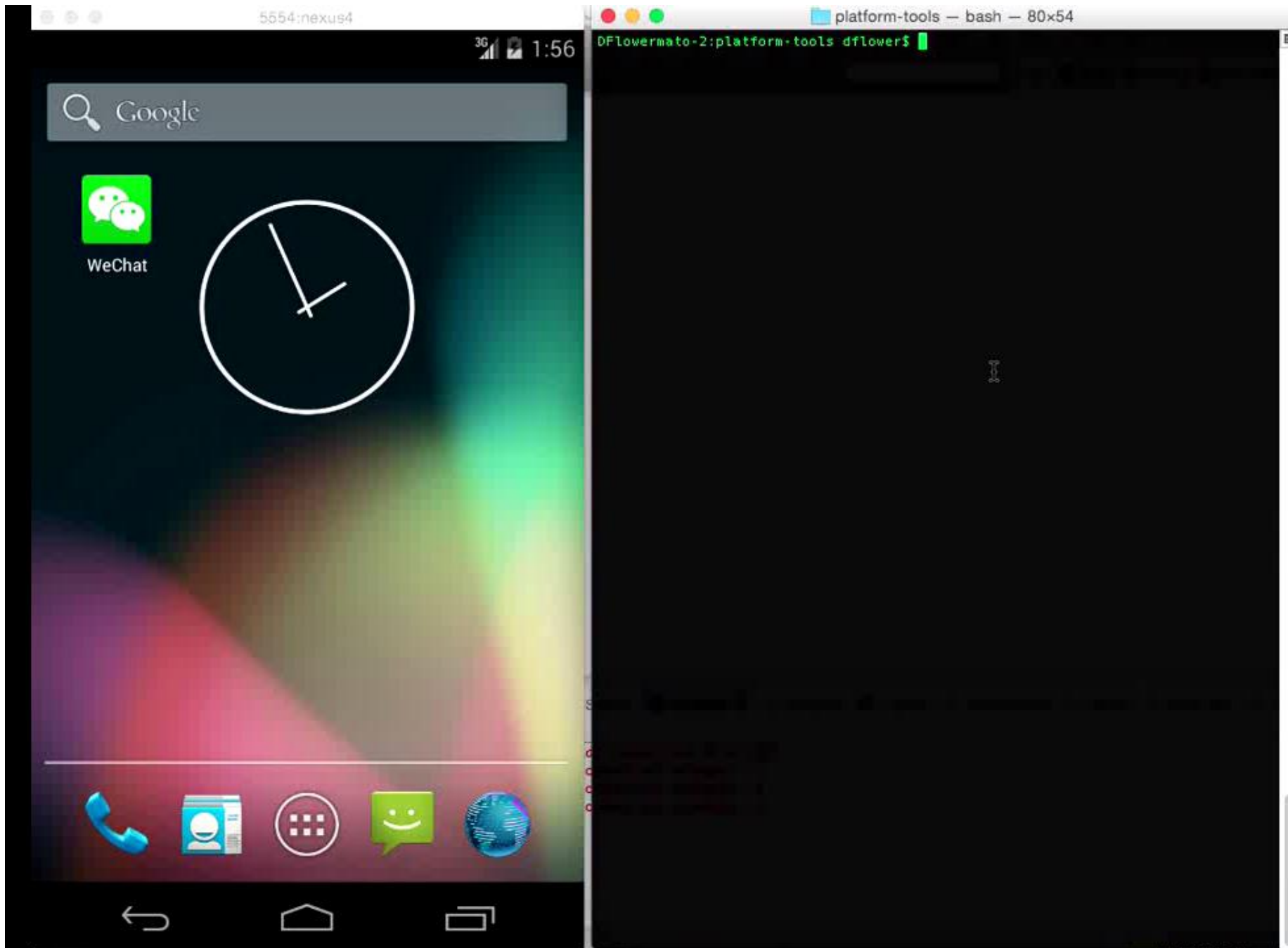
模拟用户滑动列表直至最底部，解析界面内容



2. 数据接收

DEMO : Wechat数据发送和读取

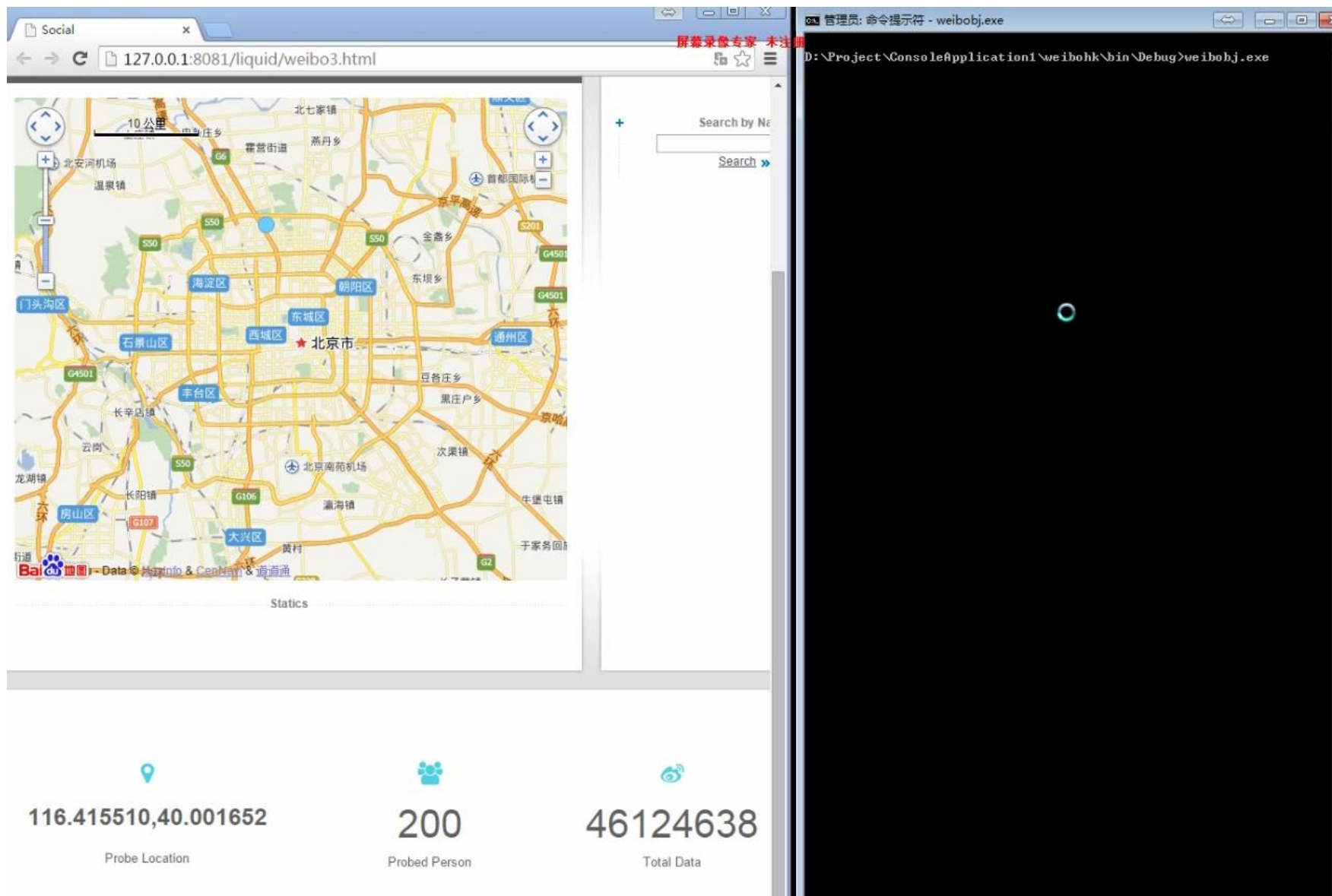
25



2. 数据接收

DEMO : Weibo数据发送和读取

26



3. 数据挖掘

- 地理位置信息展示

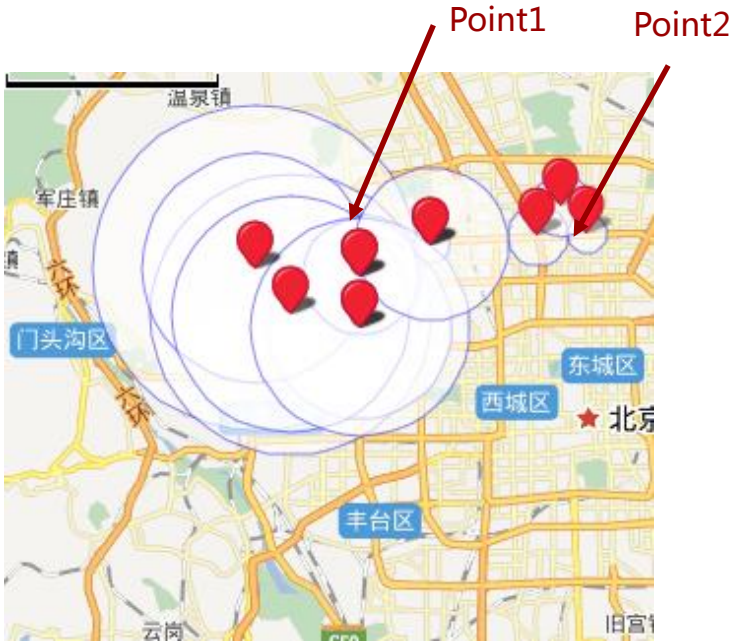
获取的数据类型	展示方式	
经纬度	坐标	
距离	以探测点为圆心，距离为半径的圆圈	可以用三点/多点定位法缩小范围



坐标展示(Weibo)



范围展示(Wechat)



三点/多点定位
缩小范围 (Momo)

小结：不同LBSN APP的安全性和利用难度比较

28

APP信息		数据安全性			利用难度		
APP	GooglePlay下载量(million)	位置精确度(APP中)	位置精确度(数据包中)	数据传输方式	爬取APP位置数据方式	利用难度	爬取到的数据精确度
Weibo	5-10	100m	1m(经纬度)，有lasttime	明文	构造数据包	低	很高
MeetMe	10-50	100m	100m(距离)	明文	构造数据包	低	中
Skout	10-50	1000m	0.01m(距离)(center: city)	明文	构造数据包(多条)	低	低
SayHi	10-50	100m	0.1m(经纬度)	明文	构造数据包	低	很高
Badoo	50-100	1000m	--	SSL单向认证(数据包参数不明)	?		中
Momo	1-5	10m	<1mm(距离)	SSL单向认证	构造数据包	低	高
Mitalk	0.5-1	100m	100m(距离)	带加密参数的明文	破解算法&构造数据包	高	中
LOVOO	10-50	0.1mile	100m(距离)有lasttime	SSL单向认证，带加密参数	破解算法&构造数据包 / 模拟器仿真	高	中
Wechat	100-500	100m	--	SSL双向认证	模拟器仿真	高	中

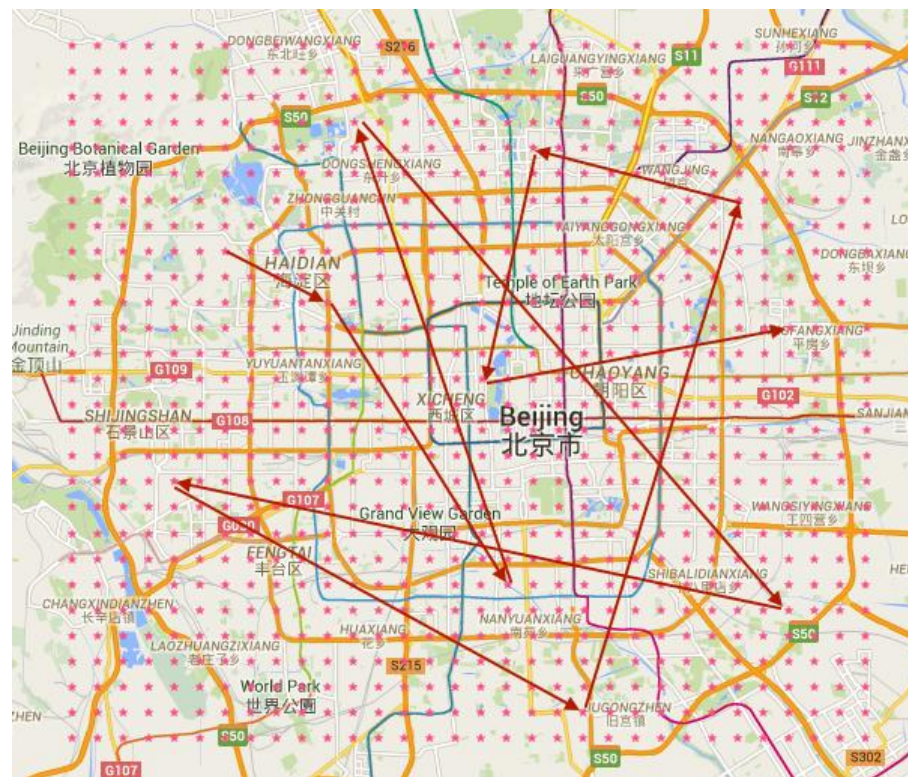
目录

- ✓ LBS APP 介绍
- ✓ 基于LBS APP进行地理位置追踪
- 实例: 追踪全北京新浪微博用户的地理位置
 - DEMO: Weibo Wall of Sheep
- 给APP厂商的建议

实例：Weibo 地理位置探测和挖掘

30

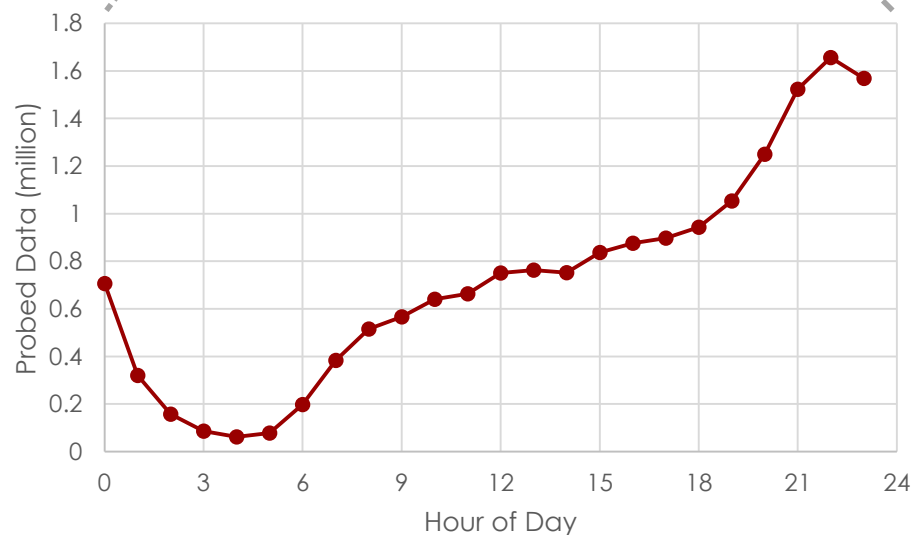
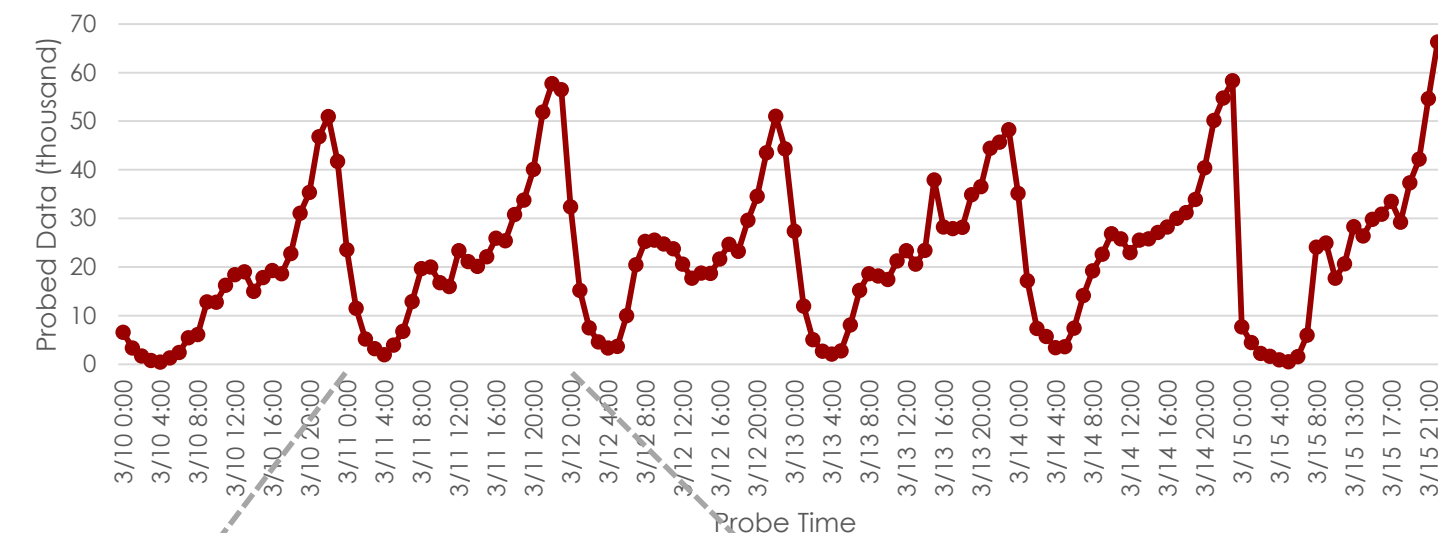
- 探测范围：北京五环内，约870km²
- 探测点：896
- 探测时间：30天/约90天
- 占用资源：1台普通配置的PC
- 探测到的数据：
 - 17,248,143 (30天)
 - 59,793,831 (约90天)
- 探测到的人数
 - 219,876 (30天)
 - 526,533 (约90天)
- 数据内容：{ 用户id/nickname, 用户坐标, 上次出现时间 }



实例：Weibo 地理位置探测和挖掘

31

- 探测数据时间分布：



用户暴露地理位置的时间

峰值：18:00 – 0:00

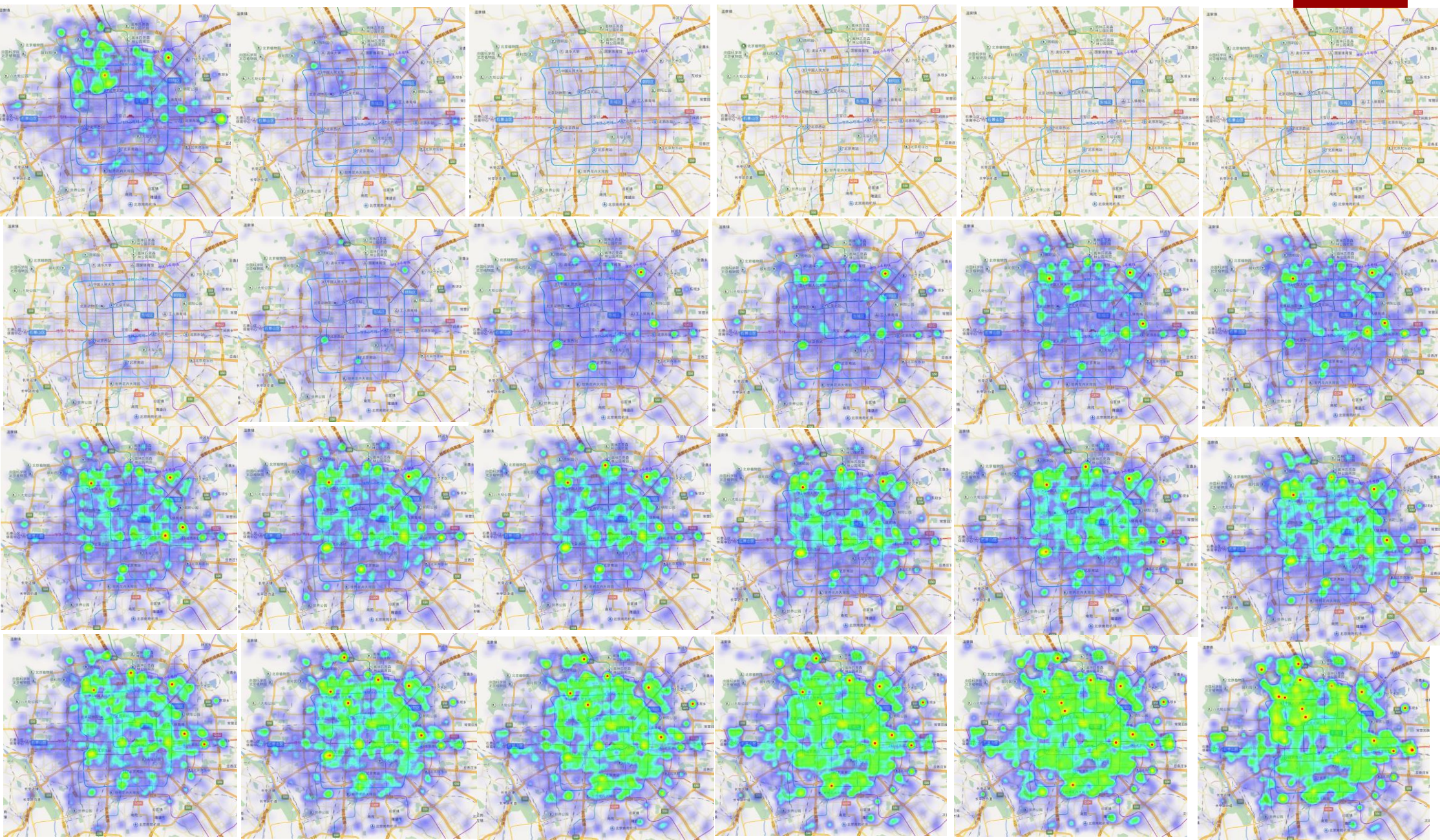
谷值：1:00 – 7:00

用户在傍晚（下班后）到午夜前（睡觉前）的社交活动比白天更为活跃

实例：Weibo 地理位置探测和挖掘

32

- 探测数据地理位置分布(0:00-24:00)



实例：Weibo 地理位置探测和挖掘

33

- 追踪用户位置

猜测：大部分用户习惯在相对固定的地理位置浏览
“周边人/周边微博”（比如在公司或家附近）

对这类用户，找到其经常暴露出的地理位置，即可对其实现追踪

实例：Weibo 地理位置探测和挖掘

34

- 追踪用户位置

为了验证上述猜测，对探测到的用户出现的位置进行聚类，
在聚类中，距离在1km之内的不同坐标点会被聚类为同一个位置

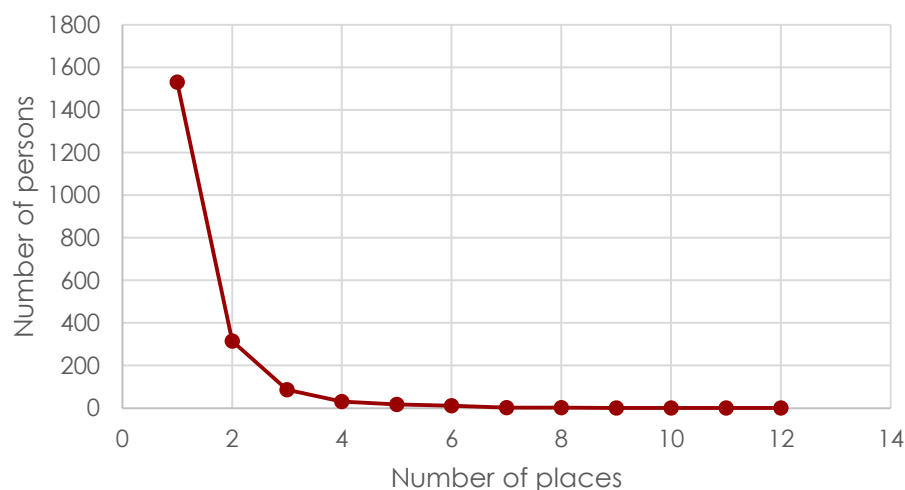
结果：

76.5%的用户：经常出现在1个位置

15.75%的用户：出现在2个位置

4.35%的用户：出现在3个位置

3.4%的用户：出现在3个以上位置



实例：Weibo 地理位置探测和挖掘

35

- 追踪用户位置

03:00~09:00 UTC(本地时间11:00~17:00)

红领巾公园 → 工作地点

12:00~15:00 UTC(本地时间20:00~22:00)

天宝源三里 (住宅区) → 家庭地点



Statics

- time:2015-03-04T03:30:52, loc:(116.49473,39.93521), place:朝阳公园
- time:2015-03-04T07:29:25, loc:(116.49483,39.93512), place:朝阳公园
- time:2015-03-04T08:40:46, loc:(116.49485,39.93512), place:朝阳公园
- time:2015-03-04T09:53:35, loc:(116.49484,39.93510), place:朝阳公园
- time:2015-03-05T02:31:54, loc:(116.49426,39.93452), place:朝阳公园
- time:2015-03-05T03:46:10, loc:(116.49470,39.93511), place:朝阳公园
- time:2015-03-05T05:30:56, loc:(116.49473,39.93521), place:朝阳公园
- time:2015-03-05T08:12:32, loc:(116.49473,39.93521), place:朝阳公园
- time:2015-03-05T08:48:10, loc:(116.49481,39.93459), place:朝阳公园
- time:2015-03-05T09:45:06, loc:(116.49481,39.93517), place:朝阳公园
- time:2015-03-05T12:05:58, loc:(116.50169,39.80116), place:亦庄
- time:2015-03-05T12:38:12, loc:(116.50173,39.80099), place:亦庄
- time:2015-03-05T13:21:33, loc:(116.50129,39.80121), place:亦庄
- time:2015-03-05T14:22:05, loc:(116.50173,39.80096), place:亦庄
- time:2015-03-05T14:38:39, loc:(116.50138,39.80120), place:亦庄

实例：Weibo 地理位置探测和挖掘

36

- 追踪用户位置：部分安全公司帐号/员工帐号追踪



360手机助手 V

北京, 东城区 <http://weibo.com/360sjzs>

360手机助手官方微博

关注 364 粉丝 988万 微博 9686



阿里云安全 V

阿里巴巴(中国)有限公司



腾讯 V

广东, 深圳 <http://weibo.com/qq>

腾讯集团 CVP高级秘书

关注 1260 粉丝 1万 微博 2176



奇虎360 V

✓ 已关注

奇虎360 职员

通过微博 weibo.com 关注

实例：Weibo 地理位置探测和挖掘

37

- 被探测到地理位置的部分大V(实名认证)帐号

个人帐号



林依轮 V

♂ 北京, 朝阳区 <http://weibo.com/yilunlin>

演员, 歌手林依轮

关注 796 粉丝 402万 微博 1578

标签: 美食 艺术 时尚 旅行 摄影



刘芸 V

♀ 北京 <http://weibo.com/liuyunliuyun>

演员

关注 390 粉丝 393万 微博 3890



cucn201白客 V + 关注

♂ 山东 泰安 <http://weibo.com/whitek>

白客, 著名的《搞笑漫画日和》系列配音员,

关注 738 | 粉丝 351万 | 微博 1756



傅盛 V

♂ 北京, 朝阳区 <http://weibo.com/fusheng7878>

猎豹移动 (原金山网络) CEO

关注 891 粉丝 301万 微博 4999

简介: 人生在世, 总要做出一些让自己觉得NB的东西。



刘家瑞Griffith V

♂ 北京, 朝阳区 <http://weibo.com/518885888>

知名演员、模特、有点影视董事长

关注 597 粉丝 47万 微博 1741

企业帐号



新浪教育 V

♀ 北京, 海淀区 <http://weibo.com/edublog>

新浪教育官方微博

关注 1787 粉丝 433万 微博 21714



新浪游戏 V

♀ 北京, 海淀区 <http://weibo.com/517656716>

新浪游戏官方微博

关注 1297 粉丝 167万 微博 34677



去哪儿网酒店控 V

♂ 北京 <http://weibo.com/qunarhotel>

去哪儿酒店搜索

关注 950 粉丝 108万 微博 13056



阿里云安全 V

♂ 浙江 <http://weibo.com/678750615> 号

阿里巴巴 (中国) 有限公司

关注 1236 粉丝 1万 微博 2260

简介: 阿里云安全集阿里巴巴集团十年攻防



360网络攻防实验室 V

♀ 北京 <http://weibo.com/360adlab>

北京奇虎科技有限公司

关注 355 粉丝 7697 微博 2103

简介: 360网络攻防实验室, 360网络

实例：Weibo 地理位置探测和挖掘

38

- DEMO：Weibo Wall of Sheep (Beijing)

Beijing - Most Fans							Beijing - Most Probed Times							Taipei - Most Fans						
#	Name	Avatar	Verify	Fans	Times Probed	Places Probed	#	Name	Avatar	Verify	Fans	Times Probed	Places Probed	#	Name	Avatar	Verify	Fans	Times Probed	Places Probed
1	陈晓东		person_V	8893933	3	3	1	向日葵BHI		None	421	396	367	1	向日葵BHI		None	421	396	367
2	董路		person_V	7263556	4	4	2	吟小吟		None	-1	329	247	2	吟小吟		None	-1	329	247
3	笑话		person_V	6628646	6	6	3	我还会再回来的-		None	178	328	144	3	我还会再回来的-		None	178	328	144
4	陈年		person_V	4371472	4	2	4	懒得起名05		None	94	281	208	4	懒得起名05		None	94	281	208
5	刘芸		person_V	4039920	21	17	5	木_雨_尘		None	-1	277	178	5	木_雨_尘		None	-1	277	178

Beijing - Most Fans							Beijing - Most Probed Times							Taipei - Most Fans						
#	Name	Avatar	Verify	Fans	Times Probed	Places Probed	#	Name	Avatar	Verify	Fans	Times Probed	Places Probed	#	Name	Avatar	Verify	Fans	Times Probed	Places Probed
1	陈晓东		person_V	8893933	3	3	1	向日葵BHI		None	421	396	367	1	向日葵BHI		None	421	396	367
2	董路		person_V	7263556	4	4	2	吟小吟		None	-1	329	247	2	吟小吟		None	-1	329	247
3	笑话		person_V	6628646	6	6	3	我还会再回来的-		None	178	328	144	3	我还会再回来的-		None	178	328	144
4	陈年		person_V	4371472	4	2	4	懒得起名05		None	94	281	208	4	懒得起名05		None	94	281	208
5	刘芸		person_V	4039920	21	17	5	木_雨_尘		None	-1	277	178	5	木_雨_尘		None	-1	277	178

实例：Weibo 地理位置探测和挖掘

39

- DEMO：Weibo Wall of Sheep (Taipei)

Beijing - Most Probed Times			Taipei - Most Fans			Taipei - Most Probed Times	
#	Name	Avatar	Verify	Fans	Times Probed	Places Probed	
1	洪宏星 adonis		person_V	1834603	17	14	
2	黄子佼		person_V	493465	3	2	
3	LOLLIPOP_F 威廉		person_V	376684	3	3	
4	猫博士		person_V	344525	7	7	
5	apple		person_V	337953	4	3	

Beijing - Most Probed Times			Taipei - Most Fans			Taipei - Most Probed Times	
#	Name	Avatar	Verify	Fans	Times Probed	Places Probed	
1	踢妮		None	18	131	106	
2	Arles5_春华 秋实		None	744	77	21	
3	忍耐__		None	18899	75	18	
4	于苗芯情		None	2465	68	58	
5	小泡姐姐儿		None	483	55	40	



THE WALL OF SHEEP

Sina Weibo Users whose Locations are Probed

地理位置暴漏的新浪微博用户

Select

Beijing - Most Fans

Beijing - Most Probed Times

Taipei - Most Fans

Taipei - Most Probed Times

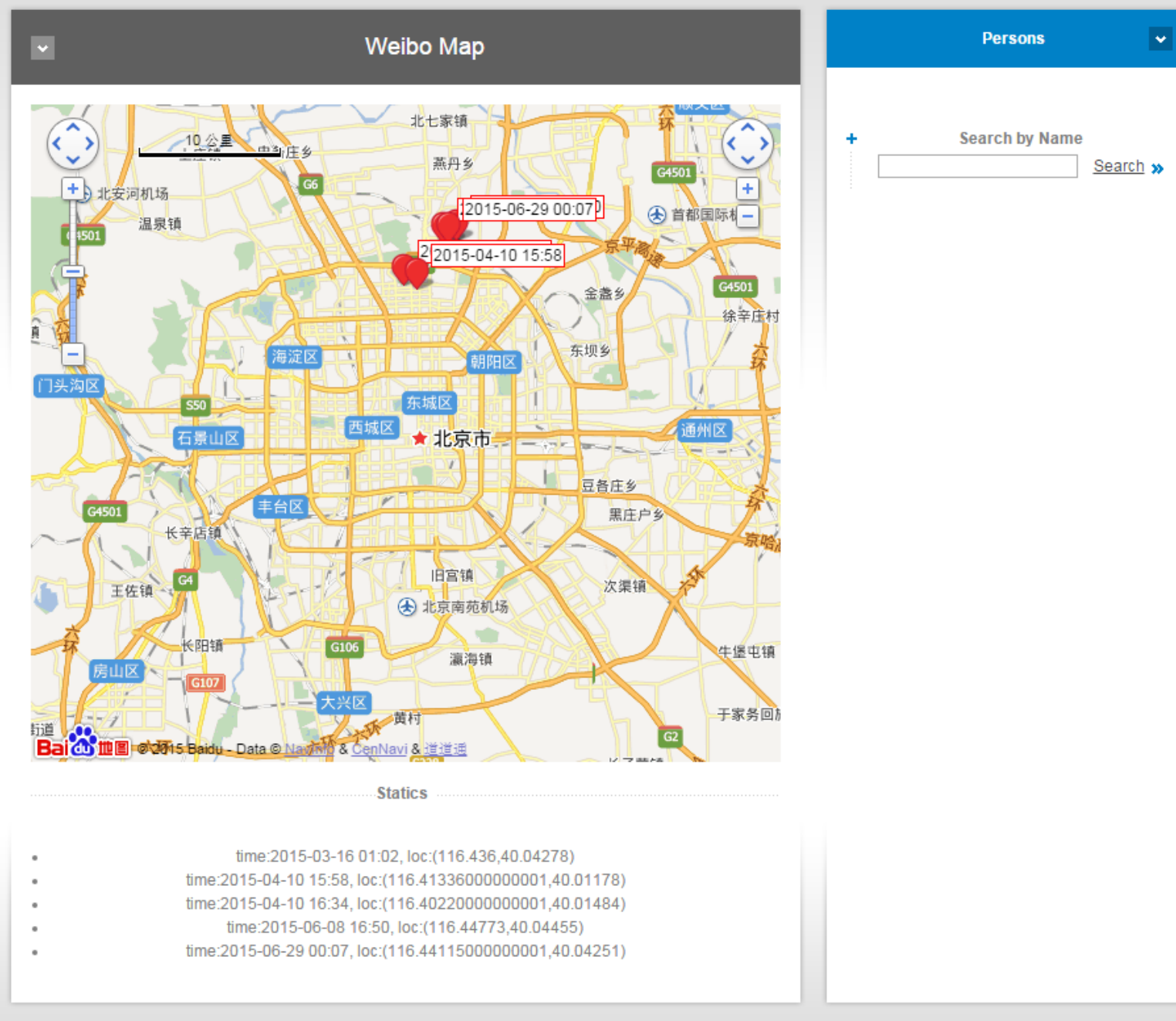
5

100

SHEEP

#	Name	Avatar	Verify	Fans	Times Probed	Places Probed	Description	ID
1	笑话		person_V	6628646	6	6	笑话，搞笑幽默博主。笑话的微博主页、个人资料、相册，笑话公司合作QQ1989575555。新浪微博，随时随地分享身边的新鲜事儿。	2814680487
2	刘芸		person_V	4039920	21	17	刘芸，演员。刘芸的微博主页、个人资料、相册，芒果娱乐。新浪微博，随时随地分享身边的新鲜事儿。	1231654104
3	手机中国联盟官博		corp_V	4004497	6	6	手机中国联盟官博，手机中国联盟官方微博。手机中国联盟官博的微博主页、个人资料、相册。新浪微博，随时随地分享身边的新鲜事儿。	2101850115

[Demo URL](#)



目录

- ✓ LBS APP 介绍
- ✓ 基于LBS APP进行地理位置追踪
- ✓ 实例: 追踪全北京新浪微博用户的地理位置
- 给APP厂商的建议

不要用明文传输！不要用明文传输！不要用明文传输！

使用双向SSL、SSL Pinning等技术防止HTTPS内容被拦截

后台服务接口应当只提供APP需要的数据，不要把所有数据都扔给前端的APP来处理

Android APP中的重要代码（如加密算法）使用Native Code或其他防破解机制

thank
you!



DFlower <dflower.zs@gmail.com>

Xiapu Luo <luoxiapu@gmail.com>