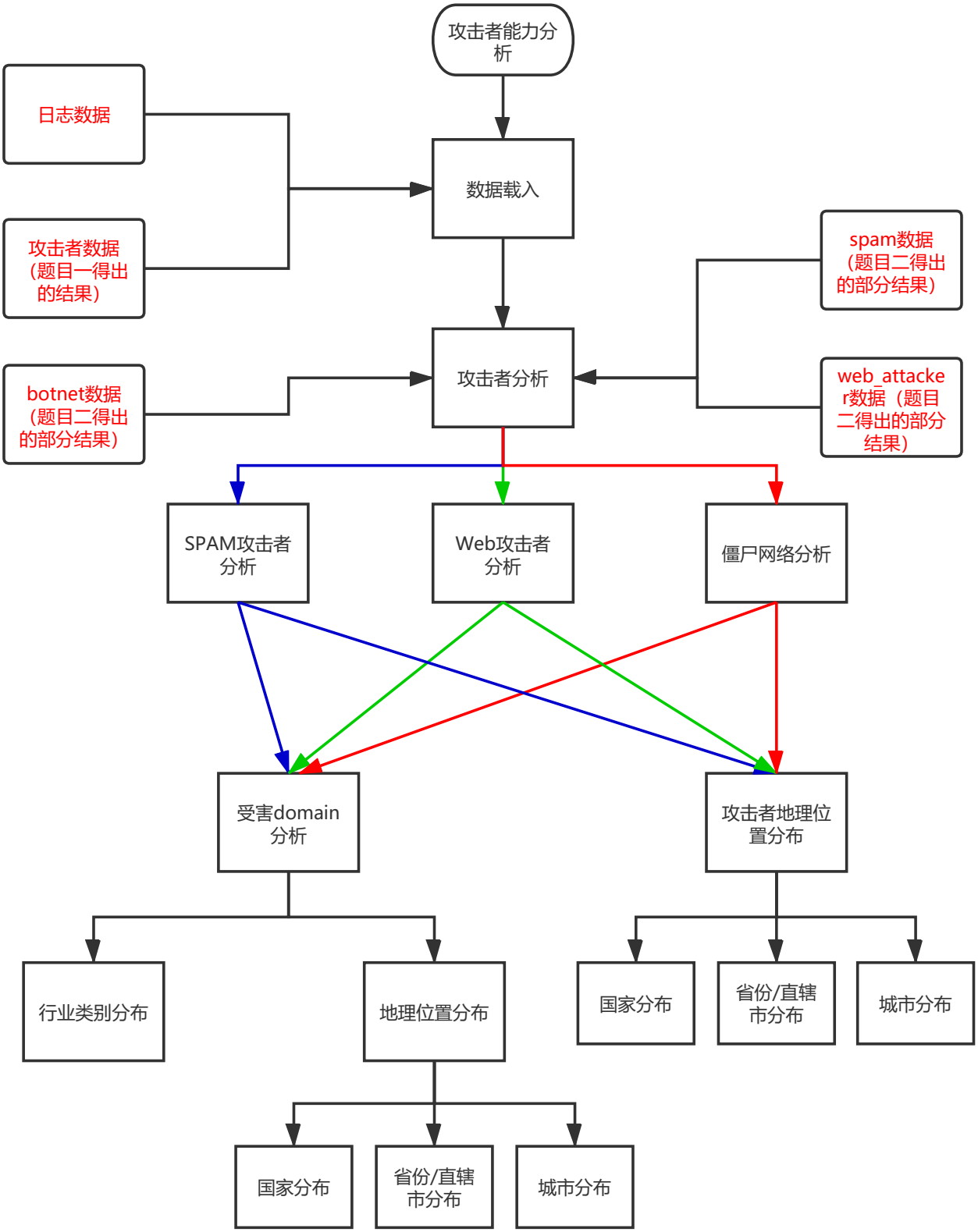
建立一套分析方法与系统，从攻击目的和攻击能力层面对攻击者进行分析

1 处理流程图

流程图如下所示，其中攻击者能力分析度量标准为：攻击域名所属行业类别的分布情况，攻击域名所在地理位置的分布情况。用到的相关数据由红色字体标注。我们将攻击者划分为三类，分别是：**SPAM攻击者**，**Web攻击者**以及**僵尸网络**，其中僵尸网络又根据DGA家族进行划分（第二题的结果）。

注：用到的额外数据打包在根目录下。分别是：

- result.csv：题目一的结果，包含我们认为是恶意攻击的IP列表
- botnet.csv：僵尸网络数据，包含了IP已经对应的DGA家族
- sd_list.txt：SPAM攻击者数据，包含了SPAM攻击者的IP
- web_attackers.csv：Web攻击者数据，包含了web攻击者的IP



2 过程描述

2.1 Spam攻击者分析

1. **数据载入**：载入日志数据，并将攻击者（题目一得到的结果）从日志数据中提取出来。

In [3]:
```
submission = pd.read_csv('result.csv')
```

In [6]:
```
ip_list = submission['file_id'].tolist()
```

In [37]:
```
ip_list
```

```
...
```

将对应的ip从日志中抽取出来

In [82]:
```
mal_list = pd.DataFrame()
for i in range(1, 32):
    print('当前迭代: %d' % i)
    if i == 26:
        continue
    if i < 10:
        path = log_path+'2018-12-0'+str(i)+'.csv'
    else:
        path = log_path+'2018-12-'+str(i)+'.csv'
    data = open(path, encoding='utf-8')
    data = pd.read_csv(data, header=None)
    # 抽取ip对应dataframe字段
#     for index, ip in enumerate(ip_list):
#         if i == 10 and index < 920:
#             continue

    mal_list = mal_list.append(data[data[0].isin(ip_list)])
    # 手动释放内存
    del data
    gc.collect()
```

```
当前迭代: 12
当前迭代: 13
当前迭代: 14
当前迭代: 15
当前迭代: 16
当前迭代: 17
当前迭代: 18
当前迭代: 19
当前迭代: 20
```

2. **数据载入**：载入domain_category数据，并将攻击日志中的domain进行类别映射

```
In [103]: load_dict
```

```
Out[103]: {'edca85768fa00dd4313fecdf6d1958e1.com': 'unknown',
           'c9607d01852c561cd498c03ec02c10ad.ru': 'unknown',
           '6d264ef6c4a84a5478d1102c659178a0.com': 'unknown',
           '00411460f7c92d2124a67ea0f4cb5f85.24f34f5e214b40f387084c5f8b40eaad.com': 'unknown',
           'aded8ca1c92227d470fc1998f9f9794a.f5b3e737510f31b88eb2d4b5d0cd2fb4.com': 'unknown',
           '4f7130958b2680204c069c324e870c60.111fad872149df74c589715b735dfe26.cn': 'unknown',
           'e389a212c2b3beb2a9a00ad2f13b8c2b.cc': 'unknown',
           'af1dcaeeddfd0fcb968f6214c421166a.io': 'unknown',
           'c93c6da9f076a3c2aea72a048a6e95c3.com': 'portal',
           'baa3894263bcead1e24604b6c4932b9d.ac1becf1fde514a37cd65d4bdd793940.org': 'unknown',
           '621e6aeb26a6a4dfe84b632c9f38ac3d.7efdfc94655a25dcea3ec85e9bb703fa.com': 'unknown',
           '3aa652f41d8b4a23e17937149c784868.713eda106b9b4e6ed773d208b4f2acc5.com': 'unknown',
           '73a844e383899ab1ee739f0b885ffb52.com': 'unknown',
           '24d587b08bb172c7d73c1f124ff8ef07.fb54f3c5992b96d001bb16e8e92d968d.f6954135cd840862b5eca39f1ba
           '6562e747c8b439b5628559e03dd57f23.com': 'unknown',
           '226b25c7503f81fb6193400ab9d97d36.net': 'portal',
           'eb8f27d2170d65a37eb72d4b005259ce.89f2f5ad765e423376f9c78100511bd7.com': 'life',
           '58d2be420b3c01f84e1889b2e32c1e5f.om': 'unknown',
           '47de7bf4f195cac50956309760038f95.cn': 'unknown',
           'e389a212c2b3beb2a9a00ad2f13b8c2b.cn': 'unknown',
```

将日志文件mal_list中的domain进行类别的映射

```
In [117]: mal_list['domain_category'] = mal_list['domain'].map(load_dict)
```

3. **domain行业类别分布**：从日志数据中提取出题目2得到的34个SPAM主机的攻击日志，假设日志中访问的domain均为受害domain，可以得到一个行业分布情况。

```
In [29]: spam_list= np.loadtxt(open("sd_list.txt",encoding='utf-8'),dtype=np.str,deliniter=None,unpack=False)
```

```
In [30]: domain_cate = collections.defaultdict(list)
         for i in spam_list:
             cate = []
             domain_cate[i] = mal_list[mal_list['ip']==i]['domain_category'].tolist()
```

```
In [34]: domain_cate
```

```
Out[34]: defaultdict(list,
                     {'103.210.237.168': [],
                      '103.213.250.5': [],
                      '103.38.43.51': [],
                      '103.38.43.74': [],
                      '103.54.60.154': ['travel',
                       'travel',
                       'travel',
                       'travel',
                       'travel',
                       'travel',
                       'travel',
                       'travel',
                       'travel',
                       'travel',
                       'travel',
                       'travel',
                       'travel',
                       'travel',
                       'travel',
```

其中，travel有449条，hospital有2条，unknown有208条。

4. **domain地理位置分布**：分析受害domain的地理位置分布，其中北京市有451条，其余208条受害domain位置为新加坡。

```
In [38]: domain_loc = collections.defaultdict(list)
         for i in spam_list:
             cate = []
             domain_loc[i] = mal_list[mal_list['ip']==i]['province(domain)'].tolist()
```

```
In [54]: domain_loc
```

```
Out[54]: defaultdict(list,
                     {'103.210.237.168': [],
                      '103.213.250.5': [],
                      '103.38.43.51': [],
                      '103.38.43.74': [],
                      '103.54.60.154': ['北京市',
                       '北京市',
                       '北京市',
                       '北京市',
                       '北京市',
                       '北京市',
                       '北京市',
                       '北京市',
                       '北京市',
                       '北京市',
                       '北京市',
                       '北京市',
```

```
In [57]: mal_list[mal_list['ip']=='113.197.36.67']
```

Out[57]:

	ip	time	domain	x-forwarded-for	country(domain)	province(domain)	city(domain)	country(ip)	province(ip)
1234669	113.197.36.67	2018-12-06T00:40:00	d41d8cd98f00b204e9800998ecf8427e.	-	NaN	NaN	NaN	新加坡	NaN
1234670	113.197.36.67	2018-12-06T01:20:00	d41d8cd98f00b204e9800998ecf8427e.	-	NaN	NaN	NaN	新加坡	NaN
1234671	113.197.36.67	2018-12-06T22:20:00	d41d8cd98f00b204e9800998ecf8427e.	-	NaN	NaN	NaN	新加坡	NaN
1234672	113.197.36.67	2018-12-06T22:15:00	d41d8cd98f00b204e9800998ecf8427e.	-	NaN	NaN	NaN	新加坡	NaN
1234673	113.197.36.67	2018-12-06T22:45:00	d41d8cd98f00b204e9800998ecf8427e.	-	NaN	NaN	NaN	新加坡	NaN
1234674	113.197.36.67	2018-12-06T22:30:00	d41d8cd98f00b204e9800998ecf8427e.	-	NaN	NaN	NaN	新加坡	NaN

2.2 Web攻击者分析

1. **载入数据**：载入web_attacker数据（题目二中得到的部分结果）

```
In [32]: web_list =  np.loadtxt(open("web_attackers.csv",encoding='utf-8'),dtype=np.str,delimiter=None,unpack=False)
```

2. **受害者domain行业分布情况**

```
In [64]: web_domain_cate = collections.defaultdict(list)
         for i in web_list:
             cate = []
             web_domain_cate[i] = mal_list[mal_list['ip']==i]['domain_category'].tolist()
```

```
In [ ]: web_domain_list = []
        for i in web_list:
            tem = dict()
            if web_domain_cate[i]:
                for cate in np.unique(web_domain_cate[i]):
                    tem[cate] = 0
                for j in web_domain_cate[i]:
                    tem[j] += 1
            web_domain_list.append(tem)
```

```
In [79]: web_domain_list
```

```
Out[79]: [{'unknown': 6},
          {'auto': 372,
           'baby': 180,
           'book': 193,
           'edu': 1675,
           'entertainment': 326,
           'finance': 3212,
           'gamble': 324,
           'games': 1212,
           'gov': 19105,
           'hospital': 326,
           'learning': 1072,
           'life': 831,
           'network_secure': 581,
           'portal': 1609,
           'sports': 139,
           'travel': 1080,
           'unknown': 8598,
           'women': 233},
          {'entertainment': 368, 'unknown': 2},
```

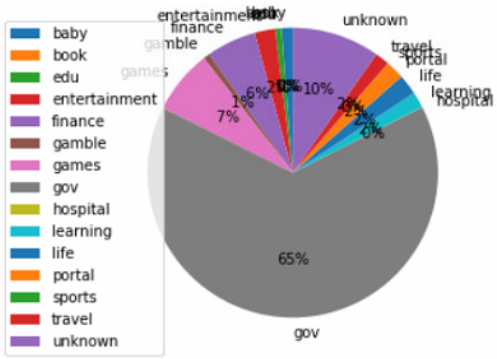
根据web_domain_list可以得出每个web攻击者攻击的受害domain行业分布。例如，分析第14个攻击者，发现其访问的domain行业分布如下：

```
In [122]: web_domain_list[13]
```

```
Out[122]: {'baby': 86,
           'book': 1,
           'edu': 49,
           'entertainment': 158,
           'finance': 383,
           'gamble': 51,
           'games': 494,
           'gov': 4499,
           'hospital': 2,
           'learning': 123,
           'life': 158,
           'portal': 143,
           'sports': 1,
           'travel': 111,
           'unknown': 678}
```

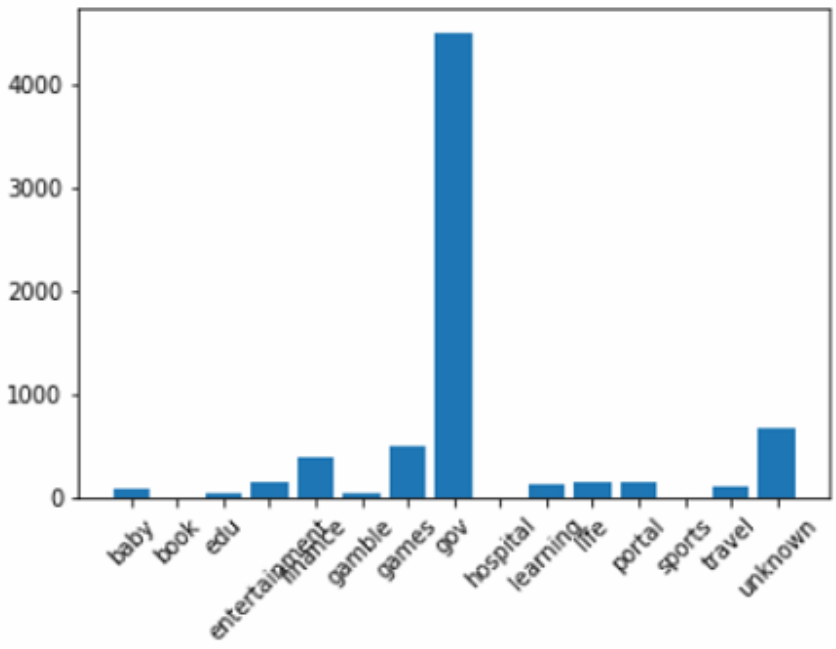
作图分析：

```
In [141]: patches, l_text, p_text = plt.pie(web_domain_list[13].values(), labels=web_domain_list[13].keys(),
                autopct='%2.0f%%',startangle=90,pctdistance=0.6)
          for t in l_text:
              t.set_size = 30
          for t in p_text:
              t.set_size = 20
              # 设置x，y轴刻度一致，这样饼图才能是圆的
          plt.axis('equal')
          plt.legend(loc='upper left', bbox_to_anchor=(-0.1, 1))
          # loc: 表示legend的位置，包括'upper right','upper left','lower right','lower left'等
          # bbox_to_anchor: 表示legend距离图形之间的距离，当出现图形与legend重叠时，可使用bbox_to_anchor进行调整legend的位置
          # 由两个参数决定，第一个参数为legend距离左边的距离，第二个参数为距离下面的距离
          plt.grid()
          plt.show()
```



其中对于'gov'类型的domain访问了4499次，且其他各行业domain都有较多的访问记录，因此认为该攻击者是一个十分活跃的攻击者，且攻击目标集中在政府部门网站。

```
In [149]: plt.bar(web_domain_list[13].keys(), web_domain_list[13].values())
          plt.xticks(rotation=45)
          plt.show()
```



3. **domain地理位置分析**

```
web_domain_loc = collections.defaultdict(list)
for i in web_list:
#     cate = []
    country = mal_list[mal_list['ip']==i]['country(domain)'].tolist()
    province = mal_list[mal_list['ip']==i]['province(domain)'].tolist()
    city = mal_list[mal_list['ip']==i]['city(domain)'].tolist()
    loc = [country, province, city]
    web_domain_loc[i] = loc
```

web_domain_loc字典：keys：web攻击者ip，values：受害domain地理位置。

2.3 僵尸网络分析

以DGA家族'Trojan'为例，进行僵尸网络的分析。

1. **载入数据**：载入僵尸网络botnet数据，并提取家族为'Trojan'的IP

```
In [17]: dga = pd.read_csv('botnet.csv')
```

```
In [21]: dga[dga['family'].isin(['Trojan'])]
```

Out[21]:

	ip	time	type	family
0	117.136.38.161	2018/9/29	NaN	Trojan
7	111.182.102.150	2018/9/26	NaN	Trojan
24	113.226.152.117	2018/9/22	NaN	Trojan
30	112.255.102.96	2017/7/1	NaN	Trojan
39	112.224.74.218	2018/10/28	NaN	Trojan
47	112.31.143.140	2018/11/30	NaN	Trojan
67	113.248.1.212	2018/11/7	NaN	Trojan
74	113.139.90.28	2018/10/22	NaN	Trojan
83	112.42.23.247	2018/8/21	NaN	Trojan
84	113.120.103.22	2018/5/27	NaN	Trojan
99	117.61.67.21	2018/5/24	NaN	Trojan
101	113.235.127.145	2018/7/21	NaN	Trojan
113	110.229.111.3	2018/6/6	NaN	Trojan
114	113.9.109.137	2018/1/16	NaN	Trojan
126	111.77.79.0	2018/7/12	NaN	Trojan
127	113.227.177.4	2018/10/28	NaN	Trojan
129	106.47.7.74	2018/11/25	NaN	Trojan
131	113.235.121.255	2018/12/3	NaN	Trojan
132	112.40.89.4	2018/10/8	NaN	Trojan

2. **从日志中抽取对应IP的信息**，这里我们丢弃了部分没有使用的列，如时间、domain等。

```
In [27]: Trojan_list = dga[dga['family'].isin(['Trojan'])]['ip'].tolist()

In [37]: Trojan_mal = mal_list[mal_list['ip'].isin(Trojan_list)]

In [40]: Trojan_mal = Trojan_mal.drop(['domain'], axis=1)

In [41]: Trojan_mal
```

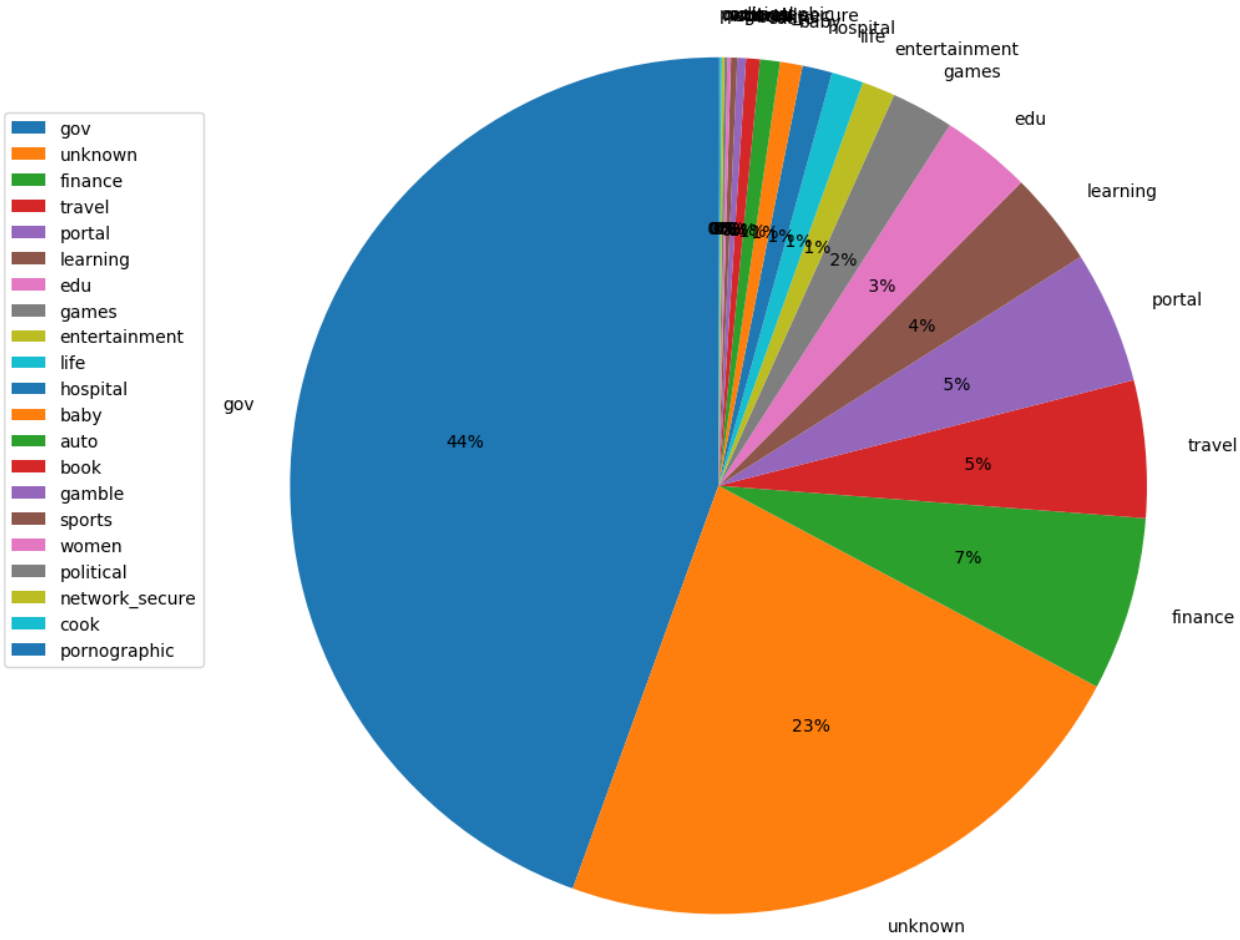
Out[41]:

	ip	country(domain)	province(domain)	city(domain)	country(ip)	province(ip)	city(ip)	domain_category
0	103.115.42.43	中国	上海市	NaN	中国	香港	NaN	unknown
1	103.115.42.43	中国	北京市	NaN	中国	香港	NaN	unknown
2	103.115.42.43	中国	北京市	NaN	中国	香港	NaN	finance
3	103.115.42.43	中国	上海市	NaN	中国	香港	NaN	gov
4	103.115.42.43	中国	广东省	深圳市	中国	香港	NaN	games
5	103.115.42.43	中国	浙江省	嘉兴市	中国	香港	NaN	gov
6	103.115.42.43	中国	江西省	吉安市	中国	香港	NaN	gov
7	103.115.42.43	中国	广西壮族自治区	南宁市	中国	香港	NaN	edu
8	103.115.42.43	中国	广西壮族自治区	南宁市	中国	香港	NaN	edu
9	103.115.42.43	中国	浙江省	嘉兴市	中国	香港	NaN	gov
10	103.115.42.43	中国	浙江省	嘉兴市	中国	香港	NaN	gov
11	103.115.42.43	中国	广东省	深圳市	中国	香港	NaN	games
12	103.115.42.43	中国	广东省	深圳市	中国	香港	NaN	games
13	103.115.42.43	中国	浙江省	嘉兴市	中国	香港	NaN	gov

3. **domain行业分布情况**

```
In [44]: Trojan_mal['domain_category'].value_counts()

Out[44]: gov              249716
         unknown          127750
         finance           36832
         travel            29071
         portal            28096
         learning          19900
         edu               19135
         games             13236
         entertainment      7003
         life               6761
         hospital           6295
         baby               4784
         auto               4212
         book               2919
         gamble             1829
         sports             1319
         women               738
         political           666
         network_secure      531
         cook                395
         pornographic        280
         Name: domain_category, dtype: int64
```



可以看到Trojan家族的攻击集中在gov和unknown两类。

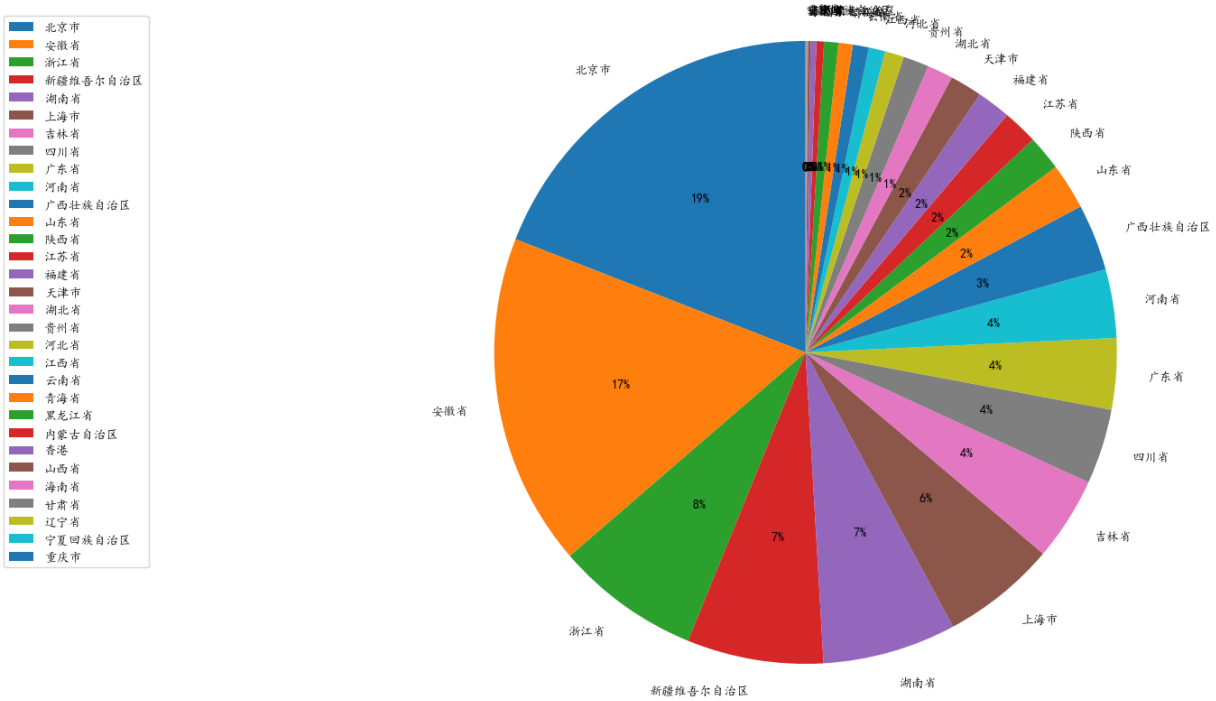
4. **domain地理位置分布情况**

国家分布：

```
In  [47]: Trojan_mal['country(domain)'].value_counts()
Out[47]: 中国     547524
         美国        566
         Name: country(domain), dtype: int64
```

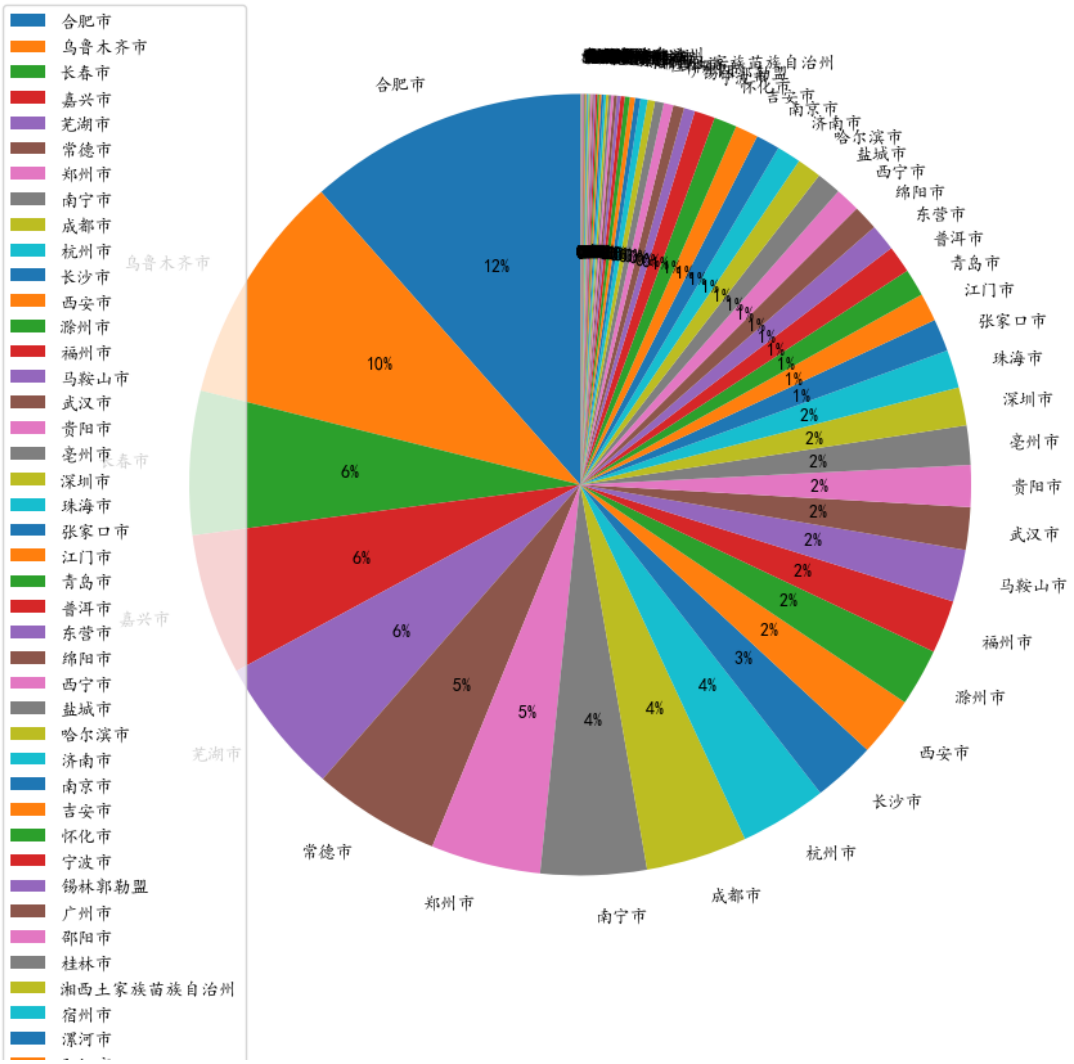
可以看到，攻击目标集中在中国。

省份/直辖市分布：



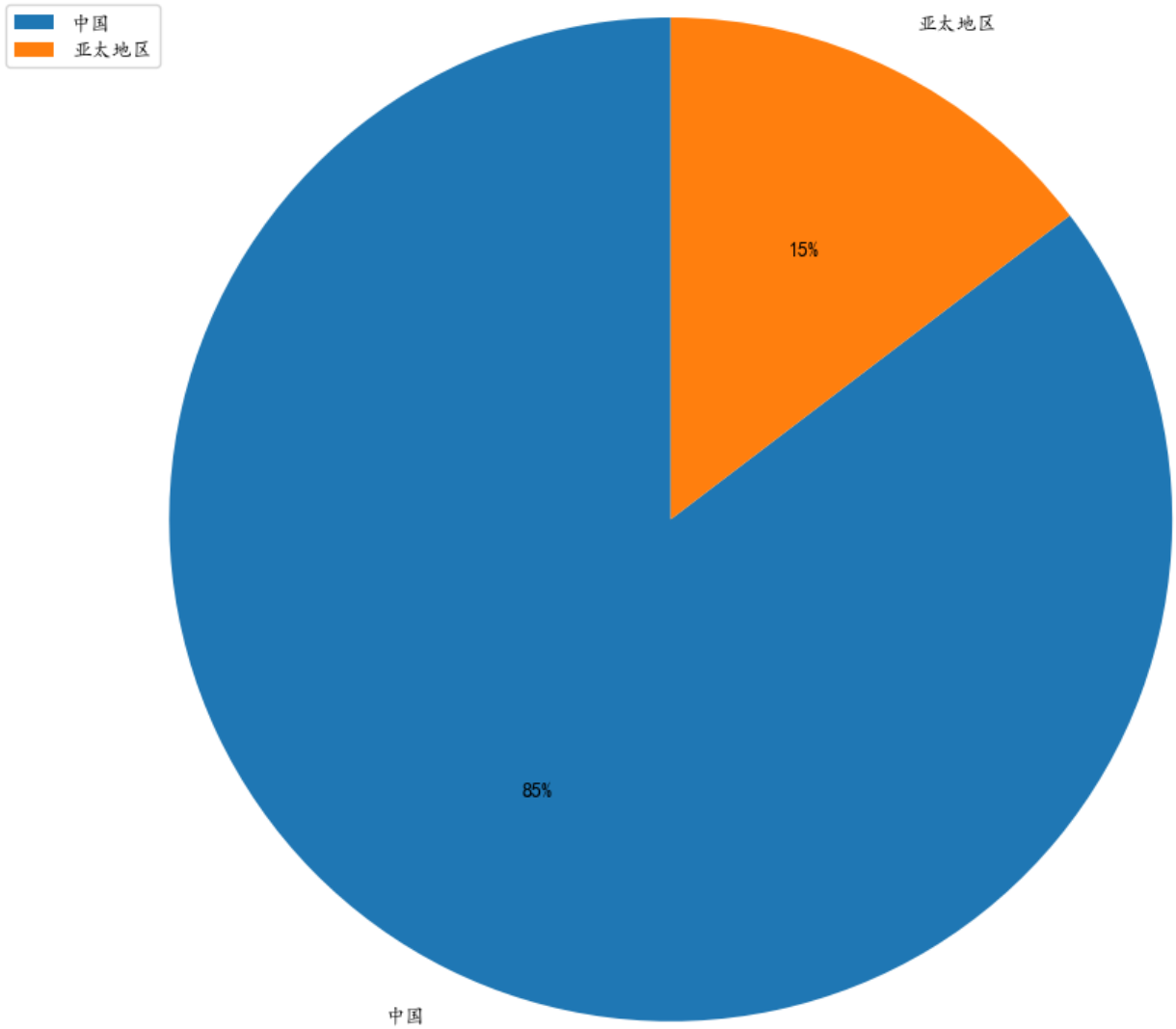
其中北京市和安徽省占比最多，其次是浙江省和新疆维吾尔自治区。

城市分布：



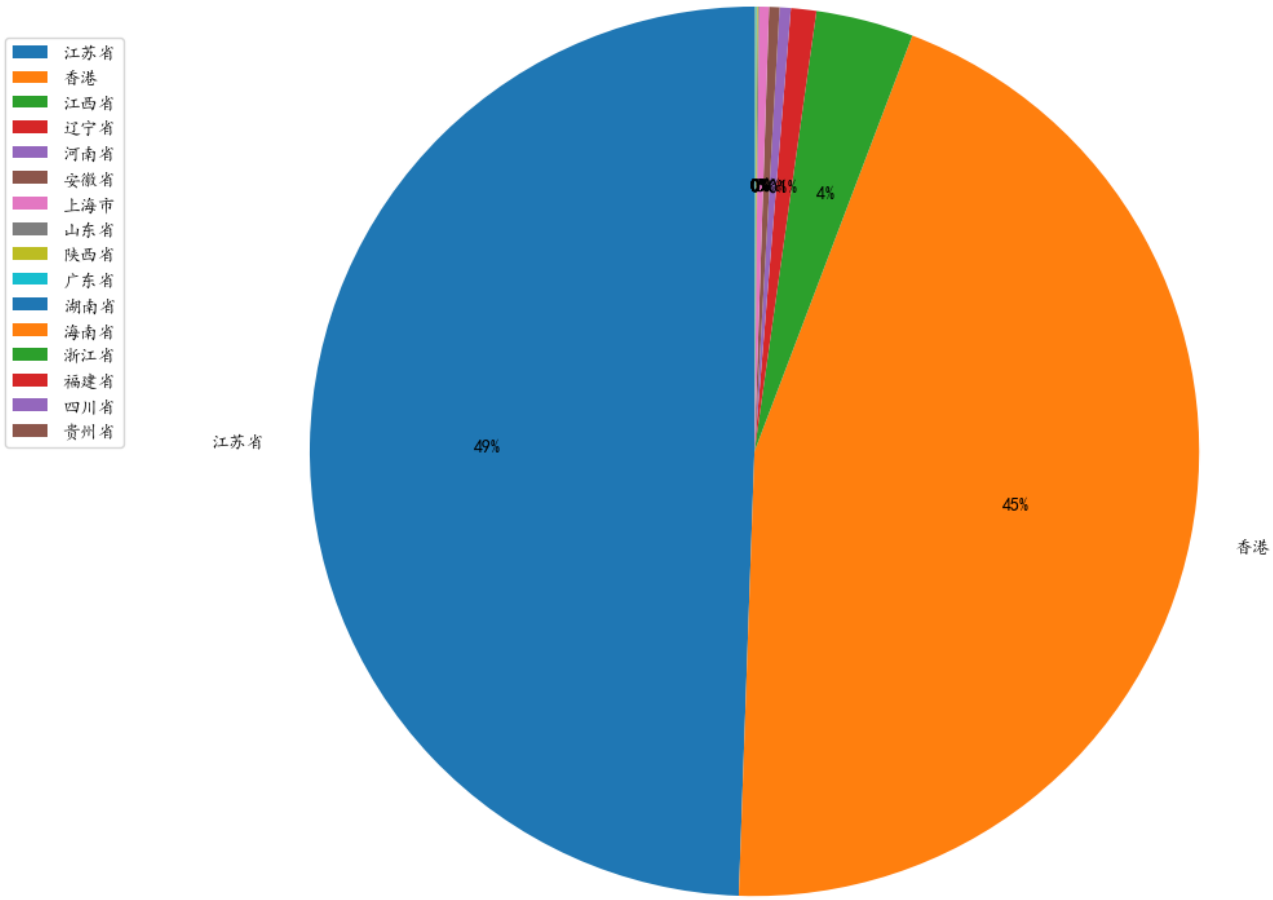
5. **僵尸主机地理位置分布情况**：分析Trojan家族中僵尸主机IP的地理位置分布，由此可以在一定程度上度量攻击者的能力。

国家分布：



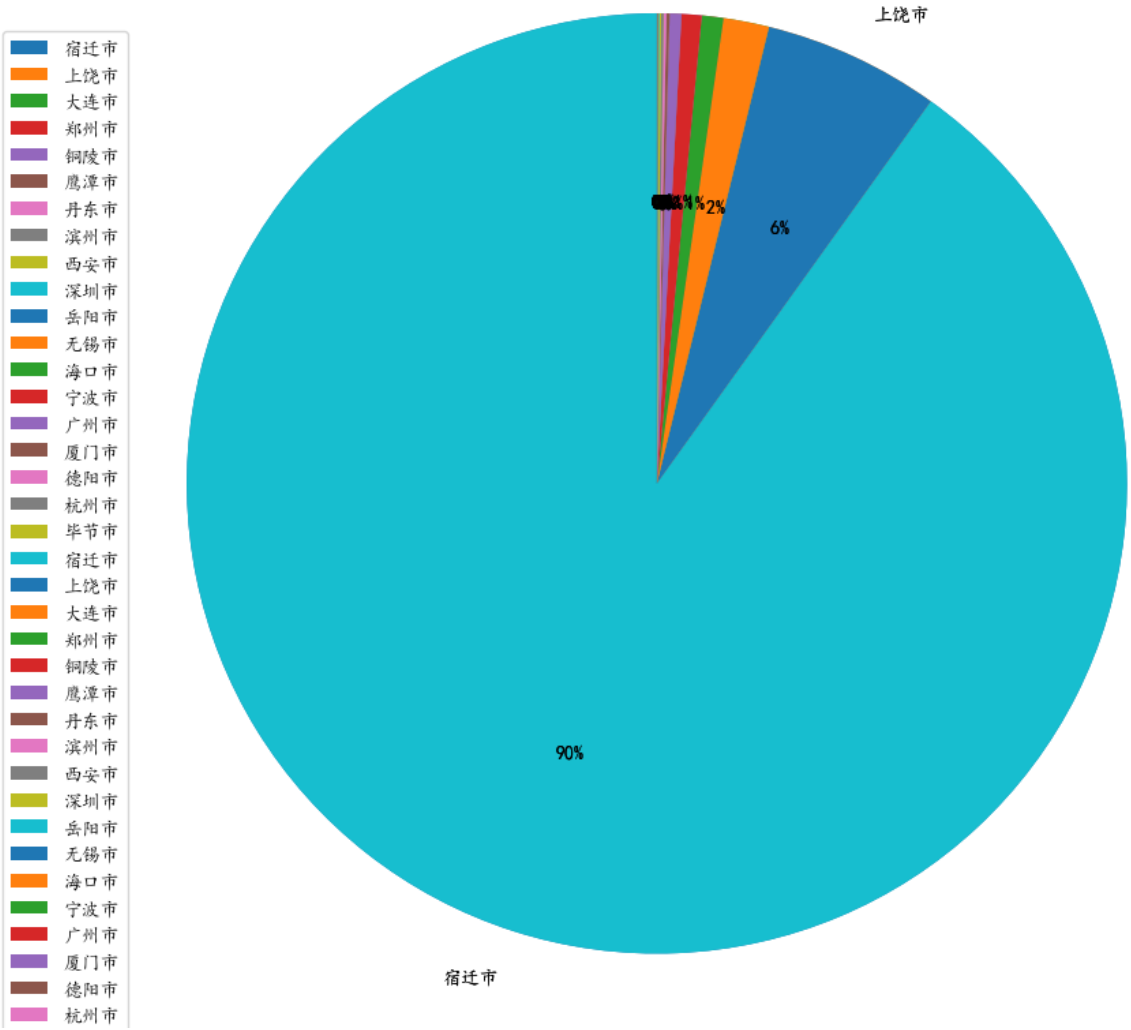
可以看到大多数僵尸主机定位在中国。

省/直辖市分布:



在省份方面，僵尸主机大多来源于江苏省和香港。

城市分布：



更进一步，在城市分布上，宿迁市占据了绝大多数。

6. **综合分析**

根据上述统计，我们大致可以得出以下结论：Trojan家族的操控者应该是国内的某个攻击者或攻击组织，其控制的僵尸主机大多来源自江苏省宿迁市以及香港。而僵尸网络攻击的目标集中在北京、安徽、浙江和新疆的政府网站。