在线实验，请到PC端体验

CDN检测插件

一、实验介绍

1.1 实验内容

这个算是扫描器中比较新颖的部分了，一些扫描器几乎没有这个功能，只有一些零散的脚本检测。

这次，我们把他集成到我们的扫描器中。

```
CDN check....
www.baidu.com [CDN Found!] Nodes:58 IP(13):111.13.100.91 14.215.177.38 220.181.1
12.143 115.239.210.27 180.97.33.107 119.75.217.109 180.97.33.108 14.215.177.37 1
11.13.100.92 61.135.169.121 220.181.111.188 61.135.169.125 220.181.112.244
```

1.2 实验知识点

- 网页抓包
- 抓包改写代码
- 获取网页返回的信息
- Chrome开发者工具的简单实用

1.3 实验环境

- Python2.7
- Xfce终端
- Sublime

1.4 适合人群

本课程难度为一般，属于初级级别课程，适合具有Python基础的用户，熟悉python基础知识加深巩固。

1.5 代码获取

你可以通过下面命令将代码下载到实验楼环境中，作为参照对比进行学习。

```
$ wget http://labfile.oss.aliyuncs.com/courses/761/shiyanlouscan8.zip
$ unzip shiyanlouscan8.zip
```

二、实验原理

CDN简介

CDN的全称是Content Delivery Network，即内容分发网络。其基本思路是尽可能避开互联网上有可能影响数据传输速度和稳定性的瓶颈和环节，使内容传输的更快、更稳定。通过在网络各处放置节点服务器所构成的在现有的互联网基础之上的一层智能虚拟网络，CDN系统能够实时地根据网络流量和各节点的连接、负载状况以及到用户的距离和响应时间等综合信息将用户的请求重新导向离用户最近的服务节点上。其目的是使用户可就近取得所需内容，解决Internet网络拥挤的状况，提高用户访问网站的响应速度。

如何检测

直接说结论吧，具体原理太长，有兴趣可以百度。我们用各地的服务器测试这个网站，如果各地得到的是不同的IP，则说明网站用了CDN。

检测CDN的用途

探测网站是否使用了CDN，如果网站使用了cdn，我们就直接停止测试，因为测试的不是源站，除非找到源站的IP。

三、开发准备

- PYTHON
- Chrome 或者使用了此内核的浏览器（360浏览器也可以）

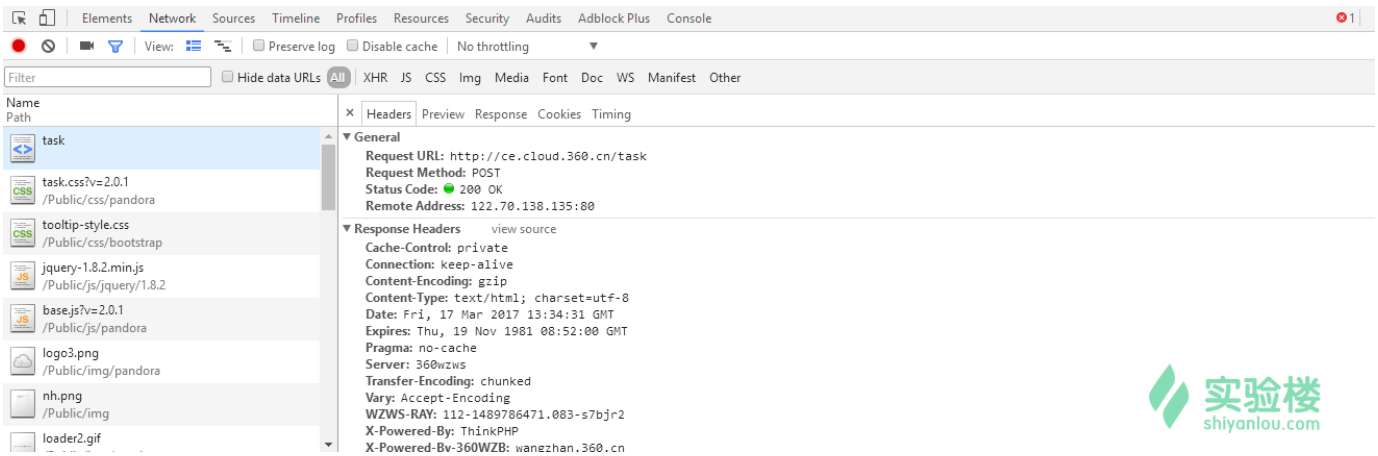
四、实验内容

4.1 抓包教程

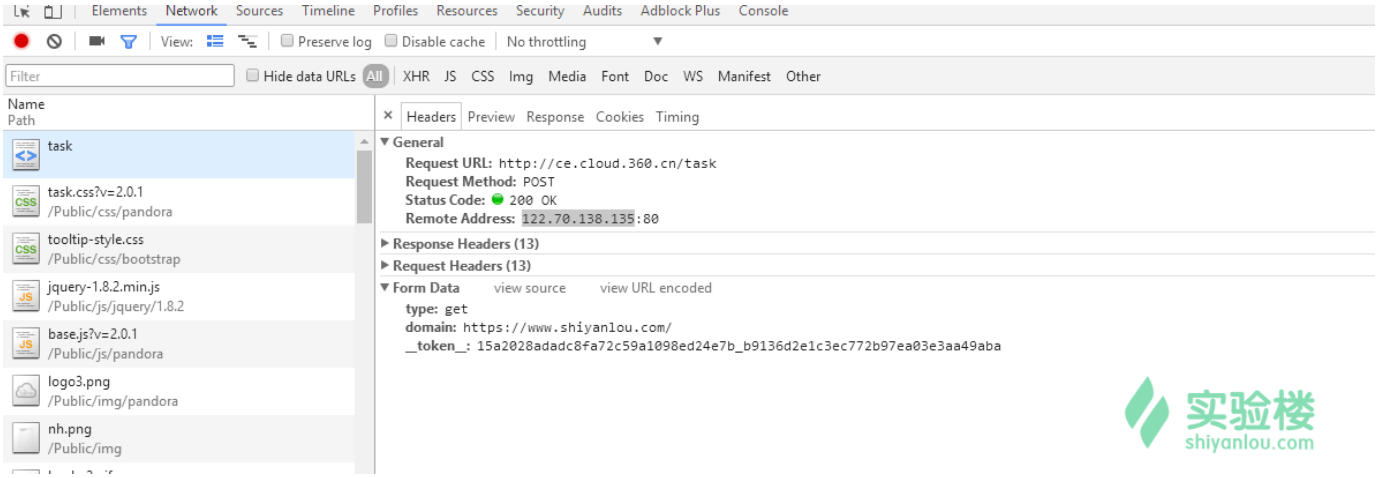
我们用http://ce.cloud.360.cn/ (http://ce.cloud.360.cn/) 这个网站进行检测
这个网站本来是一个测试网站速度的网站，但是正好可以全球PING来检测网站的IP，但是这个网站调用也比较麻烦，我们先来看看是如何抓包的。 我们用http://ce.cloud.360.cn/ (http://ce.cloud.360.cn/) 这个网站进行检测。

香港	香港	国际线路	115.29.233.149	浙江省杭州市	200	8101.44ms	1070.66ms	49.74ms	6981.04ms	38.31KB	38.31KB	5.62KB/s	查看	Ping Trace Dig 快网云主机 独立IP 58元起
河南	郑州市	电信	115.29.233.149	浙江省杭州市	200	4227.21ms	23.89ms	152.1ms	4051.22ms	38.31KB	38.31KB	9.68KB/s	查看	Ping Trace Dig 景安云服务器比域名还便宜，49元！
江苏	常州市	电信	115.29.233.149	浙江省杭州市	200	4192.55ms	10.84ms	10.14ms	4171.58ms	38.31KB	38.31KB	9.40KB/s	查看	Ping Trace Dig 常州五颜六色网络
山东	济南市	联通	115.29.233.149	浙江省杭州市	200	3458.73ms	359.08ms	156.58ms	2943.08ms	38.31KB	38.31KB	13.33KB/s	查看	Ping Trace Dig
北京	北京市	电信	115.29.233.149	浙江省杭州市	200	2704.03ms	26.34ms	29.25ms	2648.44ms	38.31KB	38.31KB	14.81KB/s	查看	Ping Trace Dig
江苏	南京市	电信	115.29.233.149	浙江省杭州市	200	7243.49ms	181.06ms	156.35ms	6906.08ms	38.31KB	38.31KB	5.68KB/s	查看	Ping Trace Dig
上海	上海市	电信	115.29.233.149	浙江省杭州市	200	7289.4ms	2.19ms	8.9ms	7278.31ms	38.31KB	38.31KB	5.39KB/s	查看	Ping Trace Dig

可以发现各地PING的IP都相同，说明没有使用CDN服务。
接下来我们进行抓包，在 Chrome 下使用 F12 。



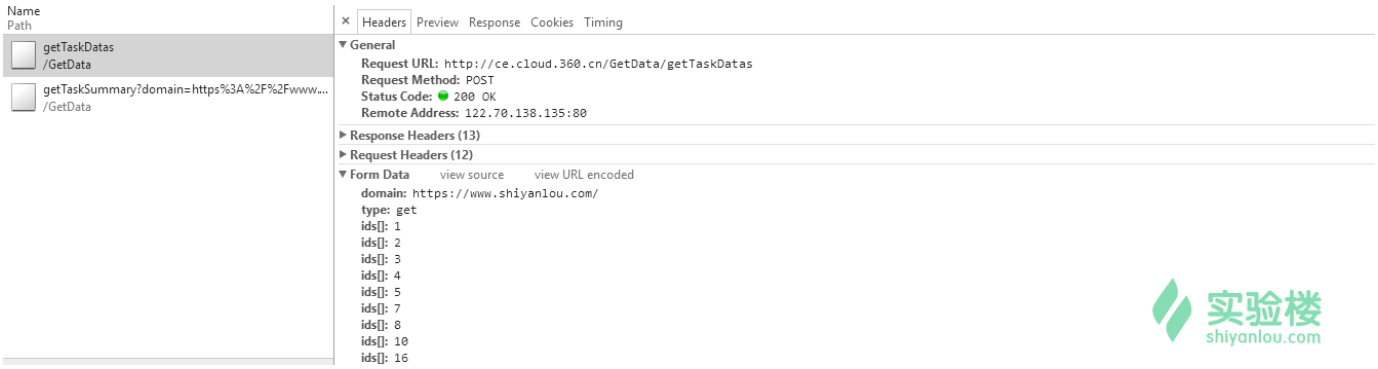
找到第一个POST包，发送的数据。



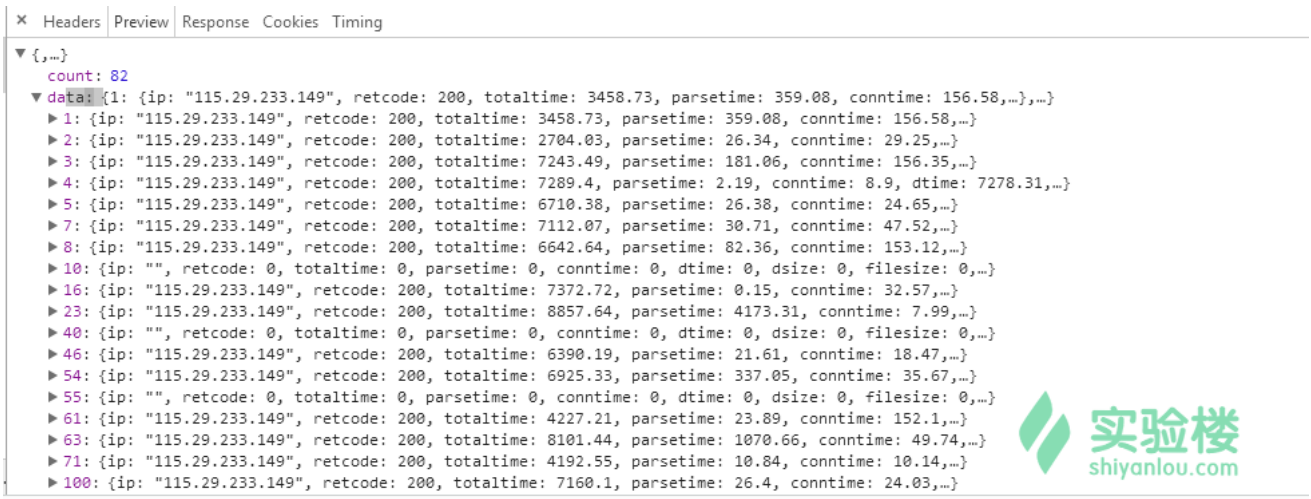
发送的数据是 domian token：

domain 是域名；
token 查看网页源码可以得到。

然后再发一个POST包得到ip列表：



可以看到会返回一个IP列表：



然后解析出返回数据中的IP地址即可。

已经有前辈写好了这个脚本，我们只需要稍微修改一下即可： https://github.com/Xyntax/POC-T/blob/2.0/script/cdn-detect.py (https://github.com/Xyntax/POC-T/blob/2.0/script/cdn-detect.py)

4.2 代码编写

在 lib/core/fun_until.py 中编写代码

先 url = urlparse.urlparse(url).netloc 解析出域名，然后先访问一下网页，来获取token，然后组合POST包用的数据。

```
data1 = _get_static_post_attr(s.get(dest).content)
data1['domain'] = url
s.post('http://ce.cloud.360.cn/GetData/getTaskDatas',data=data1)
```

_get_static_post_attr 函数原型为：

```
def _get_static_post_attr(page_content):
    """
     Get params from <input type='hidden'>

     :param page_content:html-content
     :return dict contains "hidden" parameters in <form>
     """
    _dict = {}
    # soup = BeautifulSoup(page_content, "html.parser")
    # for each in soup.find_all('input'):
    #     if 'value' in each.attrs and 'name' in each.attrs:
    #         _dict[each['name']] = each['value']
    _dict["type"] = "get"
    _dict["__token__"] = common.GetMiddleStr(page_content,'<input type="hidden" name="__token__" value="','" /></for
m>')

    return _dict
```

然后再发两个**POST**包。

```
    headers = {
        'X-Requested-With': 'XMLHttpRequest',
        'Content-Type': 'application/x-www-form-urlencoded; charset=UTF-8'
    }
    s.post('http://ce.cloud.360.cn/Tasks/detect', data=data1, headers=headers)

    time.sleep(5)  # 5 sec delay for nodes to detect

    data = 'domain=' + url + '&type=get&ids%5B%5D=1&ids%5B%5D=2&ids%5B%5D=3&ids%5B%5D=4&ids%5B%5D=5&ids%5B%5D=6&
ids%5B%5D=7&ids%5B%5D=8&ids%5B%5D=9&ids%5B%5D=16&ids%5B%5D=18&ids%5B%5D=22&ids%5B%5D=23&ids%5B%5D=41&ids%5B%5
D=45&ids%5B%5D=46&ids%5B%5D=47&ids%5B%5D=49&ids%5B%5D=50&ids%5B%5D=54&ids%5B%5D=57&ids%5B%5D=58&ids%5B%5D=61&
ids%5B%5D=62&ids%5B%5D=64&ids%5B%5D=71&ids%5B%5D=78&ids%5B%5D=79&ids%5B%5D=80&ids%5B%5D=93&ids%5B%5D=99&ids%5
B%5D=100&ids%5B%5D=101&ids%5B%5D=103&ids%5B%5D=104&ids%5B%5D=106&ids%5B%5D=110&ids%5B%5D=112&ids%5B%5D=114&id
s%5B%5D=116&ids%5B%5D=117&ids%5B%5D=118&ids%5B%5D=119&ids%5B%5D=120&ids%5B%5D=121&ids%5B%5D=122&user_ip_list
='
    r = s.post('http://ce.cloud.360.cn/GetData/getTaskDatas', data=data, headers=headers)
```

最后一个**POST**包返回的就是我们需要的数据，我们可以用**json**来解析出来，为了方便，也可以直接用正则表达式匹配出来。

```
ips = re.findall('"ip":"(.*?)"', r.content)
    ans = list(set(ips))
    msg = url

     if not len(ips):
         msg += ' [Target Unknown]'
        return msg,False

    msg += ' [CDN Found!]' if len(ans) > 1 else ''
     msg += ' Nodes:' + str(len(ips))
    msg += ' IP(%s):' % str(len(ans)) + ' '.join(ans)
```

`ans = list(set(ips))` 这个用于过滤出重复的**IP**。

4.3 完整代码

```
import requests
import re
import time
import urlparse
from lib.core import common

def _get_static_post_attr(page_content):
    """
    Get params from <input type='hidden'>

    :param page_content:html-content
    :return dict contains "hidden" parameters in <form>
    """
    _dict = {}
    # soup = BeautifulSoup(page_content, "html.parser")
    # for each in soup.find_all('input'):
    #     if 'value' in each.attrs and 'name' in each.attrs:
    #         _dict[each['name']] = each['value']
    _dict["type"] = "get"
    _dict["__token__"] = common.GetMiddleStr(page_content,'<input type="hidden" name="__token__" value="','" /></for
m>')

    return _dict

def checkCDN(url):
    """
    Detect if the website is using CDN or cloud-based web application firewall

    :param url: Target URL or Domain
    :return True / False
    """
    url = urlparse.urlparse(url).netloc

    dest = 'http://ce.cloud.360.cn/'

    s = requests.session()

    data1 = _get_static_post_attr(s.get(dest).content)
    data1['domain'] = url
    s.post('http://ce.cloud.360.cn/task', data=data1)

    headers = {
        'X-Requested-With': 'XMLHttpRequest',
        'Content-Type': 'application/x-www-form-urlencoded; charset=UTF-8'
    }
    s.post('http://ce.cloud.360.cn/Tasks/detect', data=data1, headers=headers)

    time.sleep(5)  # 5 sec delay for nodes to detect

    data = 'domain=' + url + '&type=get&ids%5B%5D=1&ids%5B%5D=2&ids%5B%5D=3&ids%5B%5D=4&ids%5B%5D=5&ids%5B%5D=6&
ids%5B%5D=7&ids%5B%5D=8&ids%5B%5D=9&ids%5B%5D=16&ids%5B%5D=18&ids%5B%5D=22&ids%5B%5D=23&ids%5B%5D=41&ids%5B%5
D=45&ids%5B%5D=46&ids%5B%5D=47&ids%5B%5D=49&ids%5B%5D=50&ids%5B%5D=54&ids%5B%5D=57&ids%5B%5D=58&ids%5B%5D=61&
ids%5B%5D=62&ids%5B%5D=64&ids%5B%5D=71&ids%5B%5D=78&ids%5B%5D=79&ids%5B%5D=80&ids%5B%5D=93&ids%5B%5D=99&ids%5
B%5D=100&ids%5B%5D=101&ids%5B%5D=103&ids%5B%5D=104&ids%5B%5D=106&ids%5B%5D=110&ids%5B%5D=112&ids%5B%5D=114&id
s%5B%5D=116&ids%5B%5D=117&ids%5B%5D=118&ids%5B%5D=119&ids%5B%5D=120&ids%5B%5D=121&ids%5B%5D=122&user_ip_list
='
    r = s.post('http://ce.cloud.360.cn/GetData/getTaskDatas', data=data, headers=headers)

    ips = re.findall('"ip":"(.*?)"', r.content)
    ans = list(set(ips))
    msg = url

    if not len(ips):
        msg += ' [Target Unknown]'
        return msg,False

    msg += ' [CDN Found!]' if len(ans) > 1 else ''
    msg += ' Nodes:' + str(len(ips))
    msg += ' IP(%s):' % str(len(ans)) + ' '.join(ans)
    return msg,True
```

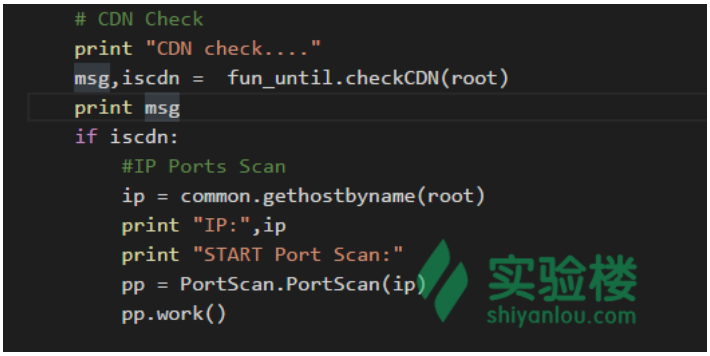
我把它放在 lib/core/fun_until.py 中

看下运行截图

```
CDN check....
www.baidu.com [CDN Found!] Nodes:58 IP(13):111.13.100.91 14.215.177.38 220.181.1
12.143 115.239.210.27 180.97.33.107 119.75.217.109 180.97.33.108 14.215.177.37 1
11.13.100.92 61.135.169.121 220.181.111.188 61.135.169.125 220.181.112.244
```

4.3 集成到扫描器中

我们编写的一些函数模块既可以单独调用，也可以把他集成到扫描器当中，代码如下图：



不要忘了在 `w8ay.py` 开头添加 `from lib.core import fun_until` 。

五、实验总结

现在。我们的扫描器运行流程是：

1. -> 检测CDN ->未发现CDN->转换ip->端口扫描
2. -> 敏感目录扫描
3. -> CMS识别
4. -> 爬虫信息收集 ->基于爬虫的各类模块

上一节 (/courses/761/labs/2671/document)
下一节 (/courses/761/labs/2673/document)

课程教师

new4

共发布过1门课程

查看老师的所有课程 > (/teacher/102428)