我的项目是一个分布式的微博爬虫。我这次的展示主要分四个部分，第一个部分是整体介绍爬虫的抓取流程以及项目的架构，第二部分是对其中的具体功能的分析，我会选择两个功能进行具体分析，第三部分是使用的主要技术以及遇到的问题第四部分是结果展示。

那我们首先看第一部分，爬虫的抓取流程以及架构，这个爬虫的架构没有采用现在网上流行的scrapy架构，采用的是request+celery+redis架构。从下面两张图我们可以看一下具体架构的区别，首先从初始URL开始，每个spider会预存初始url，spider将url交给Scheduler，Scheduler会将其交给Downloader进行下载，下载之后会交给Spider进行分析，Spider分析出来的结果有两种：一种是需要进一步抓取的链接，例如之前分析的“下一页”的链接，这些东西会被传回Scheduler；另一种是需要保存的数据，它们则被送到Item Pipeline那里，那是对数据进行后期处理（详细分析、过滤、存储等）的地方。另外，在数据流动的通道里还可以安装各种中间件，进行必要的处理。我在网上搜到这个框架的使用的流程，觉得这个框架的好处在于你只需要写爬虫的逻辑代码就可以了，具体来说就是你只要写一个继承对应模块的类（譬如说scrapy.Spider）然后重写一些方法，这个爬虫就可以运行起来了。

然后看一下这个爬虫的架构，我在这张图上用到了很多celery 的东西，譬如说broker、worker、backend。首先在celery客户端去发布任务，这里celery还支持定时任务，首先在celery客户端发布任务，然后这些任务会发送给broker（中间人），这个broker与所有可用的节点（worker）连接，broker接到任务后会把任务拆分给各个节点，各个节点根据接到的任务去具体执行，然后将任务执行的结果返回给backend模块。我这里说的任务执行的结果是指任务具体执行的返回信息譬如说执行成功失败、任务具体代码中返回的内容，而不是我在任务中需要存储的信息譬如说爬下来的微博信息，对于微博信息的存储我是用mysql存储然后存储过程写到了不同的task中。我们可以看到redis作为键值的存储系统作为worker和backend 的存储支持，这里需要用到redis是因为worker需要对任务的完成情况存储然后才可以进行下一次的任务调度。譬如说某个节点的任务执行失败然后返回给broker，broker记录任务完成情况然后再调用别的worker去执行任务。我这里用redis是因为celery在文档里建议使用redis作为第三方支持。

然后对于具体的微博爬取流程，可以看下面这张图，对于不同的爬取任务如微博个人信息爬取、粉丝爬取。首先要进行登录操作，然后再根据不同的任务进入不同网页中进行解析并提取信息的操作。

那这个爬虫的整体流程和架构就讲到这里，接下来我们就针对具体的功能也就是tasks进行分析。

首先我们讲一下登录任务，这张表展示了在未登录情况下对微博不停信息的访问情况。可以看到在不登陆的情况下有很多信息会有访问的限制，所以为了爬取这些信息，需要进行模拟登录的操作。

在分析之前我先展示登录分析最后得到的登录流程。这张图展示了未登录用户的登录操作，首先用户会在用户名输入框填写用户名，在填写用户名后，鼠标指针离开对话框后，浏览器会执行一个预登陆操作，就是向服务器发送预登陆请求来获取服务器时间、公钥等信息然后在登录请求的时候作为构造url的参数。在预登陆时会返回改账号是否需要验证码验证，所以如果需要验证码用户还需填入验证码，然后用户填完密码点击登录后，浏览器会提交登录的请求，如果账号密码正确的话，会进行一个页面转跳，转跳到对应的个人主页上。登录完成。

那我接下来再具体分析我是如何得到刚才的结论的，在具体写模拟登录功能前，我通过抓包工具将我自己人工登录微博的过程进行抓包分析，以下是得到的抓包结果。时间关系我就不一条一条的分析了，其中画红线的三处代码是我觉得挺重要的请求，分别是prelogin、login、和和302跳转页。

那首先来看prelogin 的request，作为模拟登录最重要的部分就是找到不同url构造的规律，通过模拟url来进行模拟登录。然后看这个prelogin的request的url的传参其中有su、rsakt、checkpin等变量，那如何确定这些变量都是从哪里得到的？有一个方法就是用不同账号登录并进行多次抓包然后分析这个参数的变化，如果几次抓包都不改变那么这个参数的值可能就是固定的。

经过多次抓包后我们会发现这个su的变量名是不一样的，那这个变量名是干什么的呢，其实这个是用户名经过base64编码的值，这个会在后面讲到。

然后我们去看Response，服务器返回了nonce、retcode、showpin、pubke也、servertime等参数，这些参数我们接下来会用到。

然后我们看登录的Request，发现它是一个post操作提交了一个表单，表单中的值我们刚才预登陆时得到的nonce、rsakv、servertime的值，经过多次抓包后其中su和sp的值不断变化而且也不知道从哪里得来。

那只能往下分析，发现其中有一个304转跳页面，304的意思就是本机中有的缓存文件，而且这还是个js文件，然后再浏览器里访问这个url返回了一堆JavaScript代码，我们把代码黏贴到IDE中调整代码格式，通过搜索关键字可以找到su和sp的定义

这就是在代码中搜索su的结果，可以看到su是username经过base64加密过后等到的值

这是在代码中搜索sp的结果，发现sp是password加上servertime和nonce等参数经过加密后得到的值，而且加密方式还有两种，一种是RSA加密，一种是三次sha加密。