以OTA为例，看爬虫和反爬虫大战现状

前言

爬虫与反爬虫，是一个很不阳光的行业。

这里说的不阳光，有两个含义。

第一，这个行业是隐藏在地下的，一般很少被曝光出来。很多公司对外都不会宣称自己有爬虫团队，甚至隐瞒自己有反爬虫团队的事实。这可能是出于公司战略角度来看的，与技术无关。

第二，这个行业并不是一个很积极向上的行业。很多人在这个行业摸爬滚打了多年，积攒了大量的经验，但是悲哀的发现，这些经验很难兑换成闪光的简历。面试的时候，因为双方爬虫理念或者反爬虫理念不同，也很可能互不认可，影响自己的求职之路。本来程序员就有“文人相轻”的倾向，何况理念真的大不同。

然而这就是程序员的宿命。不管这个行业有多么的不阳光，依然无法阻挡大量的人进入这个行业，因为有公司的需求。

那么，公司到底有什么样的需求，导致了我们真的需要爬虫/反爬虫呢？

反爬虫很好理解，有了爬虫我们自然要反爬虫。对于程序员来说，哪怕仅仅是出于“我就是要证明我技术比你好”的目的，也会去做。对于公司来说，意义更加重大，最少，也能降低服务器负载，光凭这一点，反爬虫就有充足的生存价值。

那么爬虫呢？

最早的爬虫起源于搜索引擎。搜索引擎是善意的爬虫，可以检索你的一切信息，并提供给其他用户访问。为此他们还专门定义了robots.txt文件，作为君子协定，这是一个双赢的局面。

然而事情很快被一些人破坏了。爬虫很快就变的不再“君子”了。

后来有了“大数据”。无数的媒体鼓吹大数据是未来的趋势，吸引了一批又一批的炮灰去创办大数据公司。这些人手头根本没有大数据，他们的数据只要用一个U盘就可以装的下，怎么好意思叫大数据呢？这么点数据根本忽悠不了投资者。于是他们开始写爬虫，拼命地爬取各个公司的数据。很快他们的数据，就无法用一个U盘装下了。这个时候终于可以休息休息，然后出去吹嘘融资啦。

然而可悲的是，大容量U盘不断地在发布。他们总是在拼命地追赶存储增加的速度。

以上是爬虫与反爬虫的历史。

爬虫反爬虫运行现状

OTA行业的爬虫与反爬虫更有趣一些，最初的爬虫需求来源于比价。

这是去哪儿的核心业务。大家如果买机票订酒店的时候，是一个价格敏感型用户的话，很可能用过他们的比价功能(真心很好用啊)。毫无悬念，他们会使用爬虫技术来爬取所有OTA的价格。他们的爬虫还是比较温柔的，对大家的服务器不会造成太大的压力。

然而，这并不意味着大家喜欢被他爬取。毕竟这对我们是不利的。于是我们需要通过技术手段来做反爬虫。

按照技术人员的想法，对方用技术怼过来，我们就要用技术怼回去，不能怂啊。这个想法是很好的，但是实际应用起来根本不是这么回事。

诚然，技术是很重要的，但是实际操作上，更重要的是套路。谁的套路更深，谁就能玩弄对方于鼓掌之中。谁的套路不行，有再好的技术，也只能被耍的团团转。这个虽然有点伤技术人员的自尊，然而，我们也不是第一天被伤自尊了。大家应该早就习惯了吧。

真实世界的爬虫比例

大家应该听过一句话吧，大概意思是说，整个互联网上大概有50%以上的流量其实是爬虫。第一次听这句话的时候，我还不是很相信，我觉得这个说法实在是太夸张了。怎么可能爬虫比人还多呢？ 爬虫毕竟只是个辅助而已。

现在做了这么久的反爬虫，我依然觉得这句话太夸张了。50%？你在逗我？就这么少的量？

举个例子，某公司，某个页面的接口，每分钟访问量是1.2万左右。这里面有多少是正常用户呢？

50%？60%？还是？

正确答案是：500以下。

也就是说，一个单独的页面，12000的访问量里，有500是正常用户，其余是爬虫。
注意，统计爬虫的时候，考虑到你不可能识别出所有的爬虫，因此，这500个用户里面，

其实还隐藏着一些爬虫。那么爬虫率大概是：

(12000-500)/12000=95.8%

这个数字你猜到了吗？

这么大的爬虫量，这么少的用户量，大家到底是在干什么？ 是什么原因导致了明明是百人级别的生意，却需要万级别的爬虫来做辅助？ 95%以上，19保1？

答案可能会相当令人喷饭。这些爬虫大部分是由于决策失误导致的。

哭笑不得的决策思路

举个例子，这个世界存在3家公司，分别提供酒店预定业务。三家公司的名字分别是A，B，C。

这个时候，客户去A公司查询了下某酒店的价格，看了下发现没有价格。于是他不打算出门了。他对整个行业的订单贡献为0。

然而A公司的后台会检测到，我们有个客户流失了，原因是他来查询了一个酒店，这个酒店我们没有价格。没关系，我去爬爬别人试试。

于是他分别爬取了B公司和C公司。

B公司的后台检测到有人来查询价格，但是呢，最终没有下单。他会认为，嗯，我们流失了一个客户。怎么办呢？

我可以爬爬看，别人有没有价格。于是他爬取了A和C。
C公司的后台检测到有人来查询价格。。。。。

过了一段时间，三家公司的服务器分别报警，访问量过高。三家公司的CTO也很纳闷，没有生成任何订单啊，怎么访问量这么高？一定是其他两家禽兽写的爬虫没有限制好频率。妈的，老子要报仇。于是分别做反爬虫，不让对方抓自己的数据。然后进一步强化自己的爬虫团队抓别人的数据。一定要做到：宁叫我抓天下人，休叫天下人抓我。

然后，做反爬虫的就要加班天天研究如何拦截爬虫。做爬虫的被拦截了，就要天天研究如何破解反爬虫策略。大家就这么把资源全都浪费在没用的地方了。直到大家合并了，才会心平气和的坐下来谈谈，都少抓点。

最近国内的公司有大量的合并，我猜这种“心平气和”应该不少吧？

爬虫反爬虫技术现状

下面我们谈谈，爬虫和反爬虫分别都是怎么做的。

为python平反

首先是爬虫。爬虫教程你到处都可以搜的到，大部分是python写的。我曾经在一篇文章提到过：用python写的爬虫是最薄弱的，因为天生并不适合破解反爬虫逻辑，因为反爬虫都是用javascript来处理。然而慢慢的，我发现这个理解有点问题（当然我如果说我当时是出于工作需要而有意黑python你们信吗。。。）。

Python的确不适合写反爬虫逻辑，但是python是一门胶水语言，他适合捆绑任何一种框架。而反爬虫策略经常会变化的翻天覆地，需要对代码进行大刀阔斧的重构，甚至重写。这种情况下，python不失为一种合适的解决方案。

举个例子，你之前是用selenium爬取对方的站点，后来你发现自己被封了，而且封锁方式十分隐蔽，完全搞不清到底是如何封的，你会怎么办？你会跟踪selenium的源码来找到出错的地方吗？

你不会。你只会换个框架，用另一种方式来爬取。然后你就把两个框架都浅尝辄止地用了下，一个都没有深入研究过。因为没等你研究好，也许人家又换方式了。你不得不再找个框架来爬取。毕竟，老板等着明天早上开会要数据呢。老板一般都是早上八九点开会，所以你七点之前必须搞定。等你厌倦了，打算换个工作的时候，简历上又只能写“了解ｎ个框架的使用”，仅此而已。

这就是爬虫工程师的宿命，爬虫工程师比外包还可怜。外包虽然不容易积累技术，但是好歹有正常上下班时间，爬虫工程师连这个权利都没有。
然而反爬虫工程师就不可怜了吗？也不是的。反爬虫有个天生的死穴，就是：误伤率。

无法绕开的误伤率

我们首先谈谈，面对对方的爬虫，你的第一反应是什么？

如果限定时间的话，大部分人给我的答案都是：封杀对方的IP。

然而，问题就出在，IP不是每人一个的。大的公司有出口IP，ISP有的时候会劫持流量让你们走代理，有的人天生喜欢挂代理，有的人为了翻墙24小时挂vpn，最坑的是，现在是移动互联网时代，你如果封了一个IP？不好意思，这是中国联通的4G网络，5分钟之前还是别人，5分钟之后就换人了哦！

因此，封IP的误伤指数最高。并且，效果又是最差的。因为现在即使是最菜的新手，也知道用代理池了。你们可以去淘宝看下，几十万的代理价值多少钱。我们就不谈到处都有的免费代理了。

也有人说：我可以扫描对方端口，如果开放了代理端口，那就意味着是个代理，我就可以封杀了呀。

事实是残酷的。我曾经封杀过一个IP，因为他开放了一个代理端口，而且是个很小众的代理端口。不出一天就有人来报事件，说我们一个分公司被拦截了。我一查IP，还真是我封的IP。我就很郁闷地问他们IT，开这个端口干什么？他说做邮件服务器啊。我说为啥要用这么奇怪的端口？他说，这不是怕别人猜出来么？我就随便取了个。

扫描端口的进阶版，还有一种方式，就是去订单库查找这个IP是否下过订单，如果没有，那么就是安全的。如果有，那就不安全。有很多网站会使用这个方法。然而这其实只是一种自欺欺人的办法而已。只需要下一单，就可以永久洗白自己的IP，天下还有比这更便宜的生意吗？

因此，封IP，以及封IP的进阶版：扫描端口再封IP，都是没用的。根本不要考虑从IP下手，因为对手会用大量的时间考虑如何躲避IP封锁，你干嘛和人家硬刚呢。这没有任何意义。

那么，下一步你会考虑到什么？

很多站点的工程师会考虑：既然没办法阻止对方，那我就让它变的不可读吧。我会用图片来渲染关键信息，比如价格。这样，人眼可见，机器识别不出来。

这个想法曾经是正确的，然而，坑爹的技术发展，带给我们一个坑爹的技术，叫机器学习。顺便带动了一个行业的迅猛发展，叫OCR。很快，识别图像就不再是任何难题了。甚至连人眼都很难识别的验证码，有的OCR都能搞定，比我肉眼识别率都高。更何况，现在有了打码平台，用资本都可以搞定，都不需要技术。

那么，下一步你会考虑什么？

这个时候，后端工程师已经没有太多的办法可以搞了。

不过后端搞不定的事情，一般都推给前端啊，前端从来都是后端搞不定问题时的背锅侠。　多少年来我们都是这么过来的。前端工程师这个时候就要勇敢地站出来了：“都不要得瑟了，来比比谁的前端知识牛逼，你牛逼我就让你爬。”

我不知道这篇文章的读者里有多少前端工程师，我只是想顺便提一下：你们以后将会是更加抢手的人才。

前端工程师的逆袭

我们知道，一个数据要显示到前端，不仅仅是后端输出就完事了，前端要做大量的事情，比如取到json之后，至少要用template转成html吧？这已经是步骤最少最简单的了。然后你总要用css渲染下吧？这也不是什么难事。

等等，你还记得自己第一次做这个事情的时候的经历吗？真的，不是什么难事吗？

有没有经历过，一个html标签拼错，或者没有闭合，导致页面错乱？一个css没弄好，导致整个页面都不知道飘到哪去了？

这些事情，你是不是很想让别人再经历一次？

这件事情充分说明了：让一个资深的前端工程师来把事情搞复杂一点，对方如果配备了资深前端工程师来破解，也需要耗费3倍以上的时间。毕竟是读别人的代码，别人写代码用了一分钟，你总是要读两分钟，然后骂一分钟吧？这已经算很少的了。如果对方没有配备前端工程师。。。那么经过一段时间，他们会成长为前端工程师。

之后，由于前端工程师的待遇比爬虫工程师稍好一些，他们很快会离职做前端，既缓解了前端人才缺口，又可以让对方缺人，重招。而他们一般是招后端做爬虫，这些人需要再接受一次折磨，再次成长为前端工程师。这不是很好的事情吗。

所以，如果你手下的爬虫工程师离职率很高，请仔细思考下，是不是自己的招聘方向有问题。

那么前端最坑爹的技术是什么呢？前端最坑爹的，也是最强大的，就是我们的：javascript。

Javascript有大量的花样可以玩，毫不夸张的说，一周换一个feature(bug)给对方学习，一年不带重样的。这个时候你就相当于一个面试官，对方要通过你的面试才行。

举个例子，Array.prototype里，有没有map啊？什么时候有啊？你说你是xx浏览器，那你这个应该是有还是应该没有啊？你说这个可以有啊？可是这个真没有啊。。那[]能不能在string里面获取字符啊？哪个浏览器可以哪个不行啊？咦你为什么支持webkit前缀啊？等等，刚刚你还支持怎么现在不支持了啊？你声明的不对啊。

这些对于前端都是简单的知识，已经习以为常了。但是对于后端来说简直就是噩梦。
然而，前端人员自己作死，研究出了一个东西，叫：nodejs。基于v8，秒杀所有的js运行。

不过nodejs实现了大量的feature，都是浏览器不存在的。你随随便便访问一些东西（比如你为什么会支持process.exit），都会把node坑的好惨好惨。而且。。。浏览器里的js，你拉到后台用nodejs跑，你是不是想到了什么安全漏洞？这个是不是叫，代码与数据混合？如果他在js里跑点恶心的代码，浏览器不支持但是node支持怎么办？

还好，爬虫工程师还有phantomjs。但是，你怎么没有定位啊？ 哈哈，你终于模拟出了定位，但是不对啊，根据我当前设置的安全策略你现在不应该能定位啊？你是怎么定出来的？连phantomjs的作者自己都维护不下去了，你真的愿意继续用吗？

当然了，最终，所有的反爬虫策略都逃不脱被破解的命运。但是这需要时间，反爬虫需要做的就是频繁发布，拖垮对方。如果对方两天可以破解你的系统，你就一天一发布，那么你就是安全的。这个系统甚至可以改名叫做“每天一道反爬题，轻轻松松学前端”。

误伤，还是误伤

这又回到了我们开始提到的“误伤率”的问题了。我们知道，发布越频繁，出问题的概率越高。那么，如何在频繁发布的情况下，还能做到少出问题呢？

此外还有一个问题，我们写了大量的“不可读代码”给对方，的确能给对方造成大量的压力，但是，这些代码我们自己也要维护啊。如果有一天忽然说，没人爬我们了，你们把代码下线掉吧。这个时候写代码的人已经不在了，你们怎么知道如何下线这些代码呢？
这两个问题我暂时不能公布我们的做法，但是大家都是聪明人，应该都是有自己的方案的，软件行业之所以忙的不得了，无非就是在折腾两件事，一个是如何将代码拆分开，一个是如何将代码合并起来。

关于误伤率，我只提一个小的tip：你可以只开启反爬虫，但是不拦截，先放着，发统计信息给自己，相当于模拟演练。等统计的差不多了，发现真的开启了也不会有什么问题，那就开启拦截或者开启造假。

这里就引发了一个问题，往往一个公司的各个频道，爬取难度是不一样的。原因就是，误伤检测这种东西与业务相关，公司的基础部门很难做出通用的。只能各个部门自己

做。甚至有的部门做了有的没做。因此引发了爬虫界一个奇葩的通用做法：如果PC页面爬不到，　就去H5试试。如果H5很麻烦，就去pc碰碰运气。

爬虫反爬虫套路现状

那么一旦有发现对方数据造假怎么办？

早期的时候，大家都是要抽查数据，通过数据来检测对方是否有造假。这个需要人工核对，成本非常高。可是那已经是洪荒时代的事情了。如果你们公司还在通过这种方式来检测，说明你们的技术还比较落伍。之前我们的竞争对手是这么干的：他们会抓取我们两次，一次是他们解密出来key之后，用正经方式来抓取，这次的结果定为A。一次是不带key，直接来抓，这次的结果定为B。根据前文描述，我们可以知道，B一定是错误的。那么如果A与B相等，说明自己中招了。这个时候会停掉爬虫，重新破解。

不要回应

所以之前有一篇关于爬虫的文章，说如何破解我们的。一直有人要我回复下。我一直觉得没什么可以回复的。

第一，反爬虫被破解了是正常的。这个世界上有个万能的爬虫手段，叫“人肉爬虫”。假设我们就是有钱，在印度开个分公司，每天雇便宜的劳动力用鼠标直接来点，你能拿我怎么办？ 第二，我们真正关心的是后续的这些套路。而我读了那篇文章，发现只是调用了selenium并且拿到了结果，就认为自己成功了。

我相信你读到这里，应该已经明白为什么我不愿意回复了。我们最重要的是工作，而不是谁打谁的脸。大家如果经常混技术社区就会发现，每天热衷于打别人脸的，一般技术都不是很好。

当然这并不代表我们技术天下第一什么的。我们每天面对大量的爬虫，还是遇到过很多高手的。就如同武侠小说里一样，高手一般都比较低调，他们默默地拿走数据，很难被发现，而且频率极低，不会影响我们的考评。你们应该明白，这是智商与情商兼具的高手了。

我们还碰到拉走我们js，砍掉无用的部分直接解出key，相当高效不拖泥带水的爬虫，一点
废请求都没有（相比某些爬虫教程，总是教你多访问写没用的url免得被发现，真的不知道高到哪里去了。这样做除了会导致机器报警，导致对方加班封锁以外，对你自己没有任何好处）。

而我们能发现这一点仅仅是是因为他低调地写了一篇博客，通篇只介绍技术，没有提任何没用的东西。

这里我只是顺便发了点小牢骚，就是希望后续不要总是有人让我回应一些关于爬虫的文章。线下我认识很多爬虫工程师，水平真的很好，也真的很低调（不然你以为我是怎么

知道如何对付爬虫的。。。），大家都是一起混的，不会产生“一定要互相打脸”的情绪。
顺便打个小广告，如果你对这个行业有兴趣，可以考虑联系HR加入我们哦。反爬虫工程师可以加入携程，爬虫工程师可以加入去哪儿。

进化

早期我们和竞争对手打的时候，双方的技术都比较初级。后来慢慢的，爬虫在升级，反爬虫也在升级。这个我们称为“进化”。我们曾经给对方放过水，来试图拖慢他们的进化速度。然而，效果不是特别理想。爬虫是否进化，取决于爬虫工程师自己的KPI，而不是反爬虫的进化速度。

后期我们和打到白热化的时候，用的技术越来越匪夷所思。举个例子，很多人会提，做反爬虫会用到canvas指纹，并认为是最高境界。其实这个东西对于反爬虫来说也只是个辅助，canvas指纹的含义是，因为不同硬件对canvas支持不同，因此你只要画一个很复杂的canvas，那么得出的image，总是存在像素级别的误差。考虑到爬虫代码都是统一的，就算起selenium，也是ghost的，因此指纹一般都是一致的，因此绕过几率非常低。

但是！这个东西天生有两个缺陷。第一是，无法验证合法性。当然了，你可以用非对称加密来保证合法，但是这个并不靠谱。其次，canvas的冲突概率非常高，远远不是作者宣称的那样，冲突率极低。也许在国外冲突是比较低，因为国外的语言比较多。但是国内公司通常是IT统一装机，无论是软件还是硬件都惊人的一致。我们测试canvas指纹的时候，在携程内部随便找了20多台机器，得出的指纹都完全一样，一丁点差别都没有。因此，有些“高级技巧”其实一点都不实用。

法律途径

此外就是大家可能都考虑过的：爬虫违法吗？能起诉对方让对方不爬吗？法务给的答案到是很干脆，可以，前提是证据。遗憾的是，这个世界上大部分的爬虫爬取数据是不会公布到自己网站的，只是用于自己的数据分析。因此，即使有一些关于爬虫的官司做为先例，并且已经打完了，依然对我们没有任何帮助。反爬虫，在对方足够低调的情况下，注定还是个技术活。

搞事情，立Flag

到了后来，我们已经不再局限于打打技术了。反爬虫的代码里我们经常埋点小彩蛋给对方，比如写点注释给对方。双方通过互相交战，频繁发布，居然聊的挺high的。

比如问问对方，北京房价是不是很高啊？ 对方回应，欧巴，我可是凭本事吃饭哦。继续问，摇到号了吗？诸如此类等等。这样的事情你来我往的，很容易动摇对方的军心，还是很有作用的。试想一下，如果你的爬虫工程师在大年三十还苦逼加班的时候，看到对方留言说自己拿到了n个月的年终奖，你觉得你的工程师，离辞职还远吗？

最后，我们终于搞出了大动作，觉得一定可以坑对方很久了。我们还特意去一家小火锅店吃了一顿，庆祝一下，准备明天上线。大家都知道，一般立flag的下场都比较惨的。两个小时的自助火锅，我们刚吃五分钟，就得到了我们投资竞争对手的消息。后面的一个多小时，团队气氛都很尴尬，谁也说不出什么话。我们组有个实习生，后来鼓足勇气问了我一个问题：

“我还能留下来吗？”
毕竟，大部分情况下，技术还是要屈服于资本的力量。

爬虫反爬虫的未来

合并之后，我们去了北京，大家坐在了一起。之前网上自称妹子的，一个个都是五大三粗的汉子，这让我们相当绝望，在场唯一的一个妹子还是我们自己带过去的（就是上面提到的实习生），感觉套路了这么久，最终还是被对方套路了。

好在，吃的喝的都很好，大家玩的还是比较high的。后续就是和平年代啦，大家不打仗了，反爬虫的逻辑扔在那做个防御，然后就开放白名单允许对方爬取了。群里经常叫的就是：xxx你怎么频率这么高，xxx你为什么这个接口没给我开放，为什么我爬的东西不对我靠你是不是把我封了啊。诸如此类的。

和平年代的反爬虫比战争年代还难做。因为战争年代，误伤率只要不是太高，公司就可以接受。和平年代大家不能搞事情，误伤率稍稍多一点，就会有人叫：好好的不赚钱，瞎搞什么搞。此外，战争年代只要不拦截用户，就不算误伤。和平年代还要考虑白名单，拦截了合作伙伴也是误伤。因此各方面会更保守一些。不过，总体来说还是和平年代比较happy。毕竟，谁会喜欢没事加班玩呢。

然而和平持续的不是很久，很快就有了新的竞争对手选择爬虫来与我们打，具体是谁你们应该也猜得到。毕竟，这是一个利益驱使的世界。只要有大量的利润，资本家就会杀人放火，这不是我们这些技术人员可以决定的。我们希望天下无虫，但是我们又有什么权利呢。

好在，这样可以催生更多的职位，顺便提高大家的身价，也算是个好事情吧。