0x01 问题背景

目前红队工具十分的多，开发者也不同，工具调用的方法也不一样，其实各个工具之间或多或少都有自己的长处，但是也和其他同类型的工具有功能交集，即功能重叠的地方。

在日常红队项目中，经常会遇到一个问题，就是一个工具，其实只需要用他一部分功能，但是基于某个特定的环境，这个时候又需要增加一些新功能，但是工具原本却并不具备这个功能。

这里就存在一个快速开发的问题，即有没有办法能够抽取市面上已经有的工具的现有功能，然后快速开发，实现我们要的新功能，让功能快速的跑起来。

0x02 构想中的解决方法

解决方法层面个人认为可能涉及到两个层面。

1.真正要解决什么问题

2.怎么解决这个问题

这里先谈第二点，即怎么解决问题。

这里我认为其实可以用累积木的方法来解决这个问题。

即，在积木游戏中，积木都是由最基本的块组成的。

然后一些基本的块，又可以汇聚成一个模块，比如一个门，一个小球，一个座子之类的。

然后假设现在我们要盖一个房子，那么现在就需要一个屋顶，一个地基座子，一些墙，一些门之类的东西。

我们就可以从现有的模块中直接抽取，然后组合成我们想要的房子，并不需要自己再从头开始造这个房子。

写代码也是如此，实战中时间紧张，遇到问题最直接的办法就是直接缝合。

即，抽取具有对应功能的几个工具，然后也不用改代码，直接看下输入输出，中间自己写一些小脚本，对不同工具的输入输出做处理，就可以直接缝合起来了。

也就是变成了管道，变成了程序流。

这个思维感觉很基本，本来没什么好说的，但是自己实战中一直在用，同时也不断地看别人的文章以及和别人交流，最后发现其实很多红队或多或少都下意识的在用这个方法，只是没有写出来，所以这里就写一下。

然后这里也可以谈到Cobalt Strike的开发，最初的CS开发者的思维也是如此，CS只完成基本的功能，其他例如免杀之类的东西，都先不管，因为免杀这玩意，今天再牛逼的技术，一公开，明天就不免杀了，团队一直维护持续免杀，是十分消耗精力的，不如交给攻击者自己去完成。

老CS开发者的思维就是典型的积木思维，因为免杀那些玩意都可以用插件的办法来解决，灵活多变，不用硬嵌到代码里面。

CS搞一个基本的架子，然后其他要用啥，就用插件直接插进去，一个基本的框架就可以因为插件的不同，形成具有个性化的工具。

那么再回到了点1，真正要解决什么问题。

这里拿一个打点的问题往回递推来举例子。

打点就那么几个手法，最后的本质结果无非就是RCE，能上线，然后通过这台机器能打内网。

那么根据这个结果往回递推，发现有些手法可以做到RCE，比如XPCMDSHELL可以执行命令，那么这个可以作为一个打点的手法，那么再往回推，我们可能就会去网站的后台找数据库执行功能，那么怎么进入网站后台呢，我们可能就会再去找网站的弱口令或者信息泄露，看看有没有账号密码能够登录到后台。

根据结果往回去回溯是我经常用的一个方法，这样做起事情来不是很迷茫，进度在心里也有个把控，也知道自己在干嘛。

那么还是说打点。

我们把所有的RCE手法汇聚一下，无非就是钓鱼，反序列化，SQL注入，命令注入，文件上传等几个手法，其实也不多。

然后根据上面的手法往回推导，怎么才能触发这些漏洞，就可以找到路径。

比如反序列化，有些已知框架存在反序列化，因此我们会去研发工具来检测。

有些反序列化存在的包我们一个个去看太麻烦了，比如Fastjson，因此我们会去研发Burp插件来自动检测Fastjson反序列化，节省人力。

然后我们发现漏洞其实非常多，有没有一个工具能够直接一键检测常用RCE漏洞，于是就有了Xray。

往回推导的本质，是需求决定开发什么功能。

线性推导，是感性觉得需要什么，然后开发什么功能。

可能很多时候就是自己擅长什么技术，然后做了一个东西出来，但是最后的应用场景却很难找到。

往回推导就能避免这个问题，同时做起事情来不太会迷茫，每一个东西搞出来都知道最后能干什么，在往哪个方向上走。

因为红队安全研究本质我认为还是应用学科的研究思路，和科学家那个体系不太一样，本质就是实战中找思路的一个学科，脱离实战去研究问题肯定不行，因为脱离了实战，就脱离了对抗的本质，你都不知道人家是怎么防守的，怎么做针对性绕过，最后的攻击结果怎么会好。

防守方也是一样，做一个防守策略出来，都没有经过攻击方的检验，怎么敢说是有效的。

两个思路，分别解决两个问题。

一个是解决方向问题，一个是解决执行的时候的效率问题，不单单是改工具的时候用这个思路，自己开发工具的时候也是这个思路，因为红队的面很广，外网，内网，免杀，java安全等，其实很多时候都会用到相同的功能模块，撇开语言特性不谈，在写通用工具的时候，比如用Golang写，很多东西都应该做成单独的模块化的东西，把输入输出写好，就好像积木的接口一样，然后用的时候要用什么直接拼一下，功能不用再写了，最后写写接口转换器，直接就成了，实战中应对更加多变的环境的时候可以快速开发出自己想要的工具。

Done