**冰蝎v4.0**

简介

在冰蝎的最近更新中,添加或修改了如下一些功能

1. 修复了在zimbra环境下的兼容性问题；2.修复了在exchange环境下的兼容性问题；3.修复了Linux环境下打开文件失败的问题；4.修复了命令执行中输入反斜杠导致后续无法输入新命令的问题；5.修复了列目录时目录跳动的问题；6.修复JDK18+执行命令乱码的问题；7.修复内网穿透模块的几个影响隧道稳定性的几个问题；8.修复了代码编辑框的复制粘贴问题；9.其他一些优化；新增：1.新增支持多线程超大文件上传、下载；2.新增文件打包压缩；3.新增数据库连接配置可保存；4.取消硬编码通信协议，传输协议完全自定义，并支持即时在线校验测试。5.新增平行世界模块，可对目标内网资产进行管理；6.新增主机扫描、端口扫描、服务识别模块；7.新增支持Java9~java15+版本Agent内存马一键注入；8.新增支持Java Agent无文件落地注入内存马；9.新增多层网络子Shell穿透模块，实现多层”蝎中蝎”；10.新增离线模式，自动缓存数据，如shell丢失，可离线查看已缓存内容；11.开放插件开发模块，可开发自定义插件，内置多款插件；12.支持二进制服务端，服务端不再依赖web。

特征分析

php抓包

看包没有发现什么特征，但是可以发现它是POST请求的



Accept

- Accept 在默认情况下都为`application/json, text/javascript, \*/\*; q=0.01`属于弱特征



Accept-Encoding:

同样的,在用户默认不提供的情况下, Accept-Encoding:默认为Accept-Encoding: identity



我们日常使用的浏览器中常见的Accept-Encoding有Accept-Encoding: compress, gzip //支持compress 和gzip类型Accept-Encoding:　//默认是identityAccept-Encoding: \*　//支持所有类型 Accept-Encoding: compress;q=0.5, gzip;q=1.0

Accept-Encoding: gzip;q=1.0, identity; q=0.5, \*;q=0 // 按顺序支持 gzip , identitychrome下的Accept-Encoding: gzip, deflate

因为identity是默认但一般情况下是置空的缘故,可以作为弱特征来使用

Accept-Language



在使用chrome时这个值通常为.或者为空,在面对这种设置时,可以将其看作为弱特征

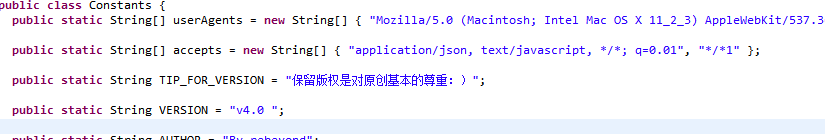
user-Agent

该特征属于弱特征,且在4.0中有个改进,做检测及过滤的时候要多加注意。

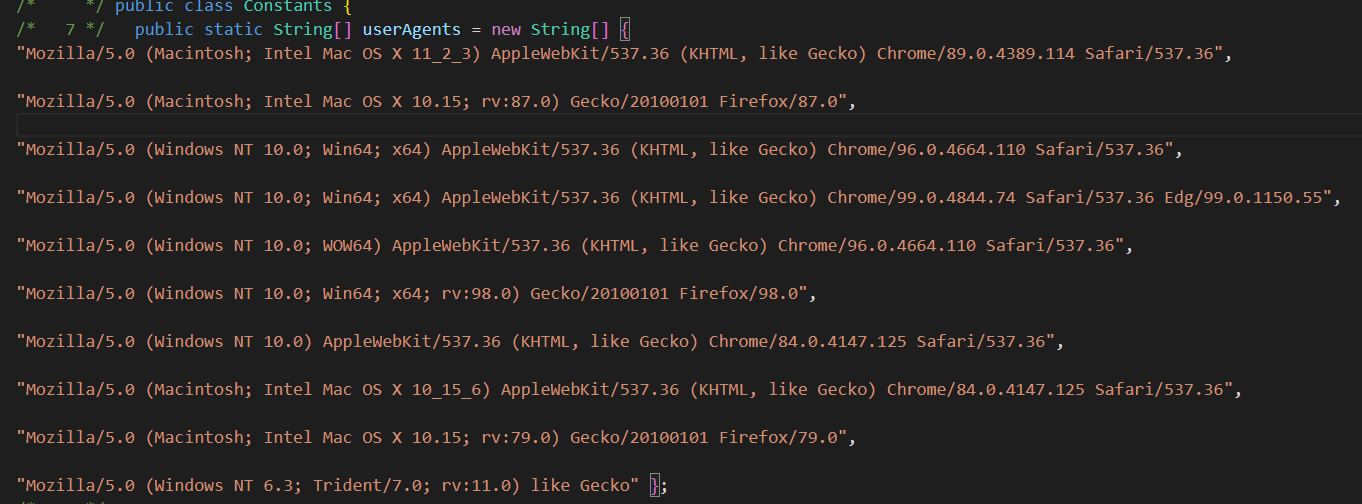
冰蝎.4.0 相比较于3.0在这方面做出了很大的改进,对于3.0使用的user-Agent头都是属于N年前的浏览器产品。现实生活中很少有人使用,在4.0中将user-Agent都设置为了近代来比较流行的浏览器头

每次请求都会随机选择一个user-Agent。但是如果用户默认不提供ua头，则从系统中随机选择一个ua头。

通过反汇编拿到ua数据



这个也可以作为waf规则特征。



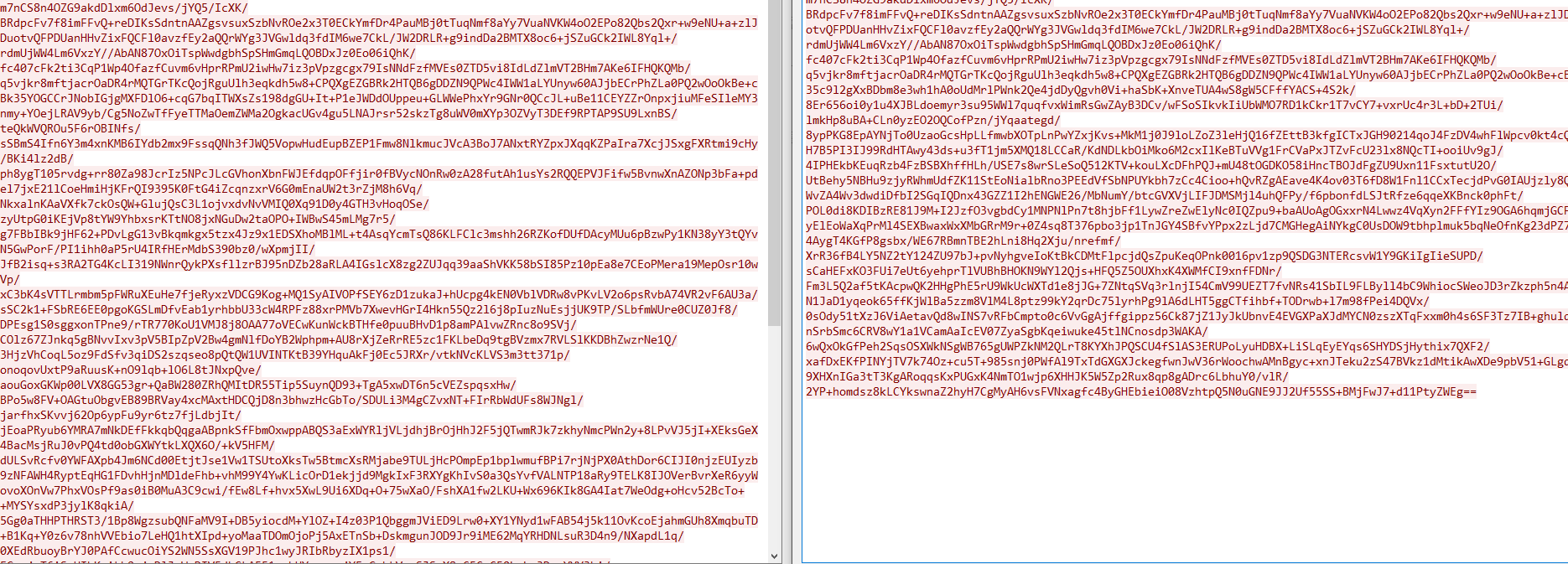
遗憾的是,冰蝎4.0相比较于冰蝎3.0,冰蝎4.0的请求内容更加强大且复杂

在冰蝎3.0中虽然对请求的内容加密，但是被加密的内容中，并没有随机填充的部分。下以无密钥交互为例子.



在冰蝎3.0中，任何请求，最终都会调用Utils.getData函数，对请求的参数加密。对于上传文件，命令执行来讲，加密的参数不定长。但是对于密钥交互，获取基本信息来讲，payload都为定长，且无随机padding。

但在冰蝎4.0中. 被加密的内容的内容都是长度不定的, payload的长度及内容都很随机,不能作为检测依据



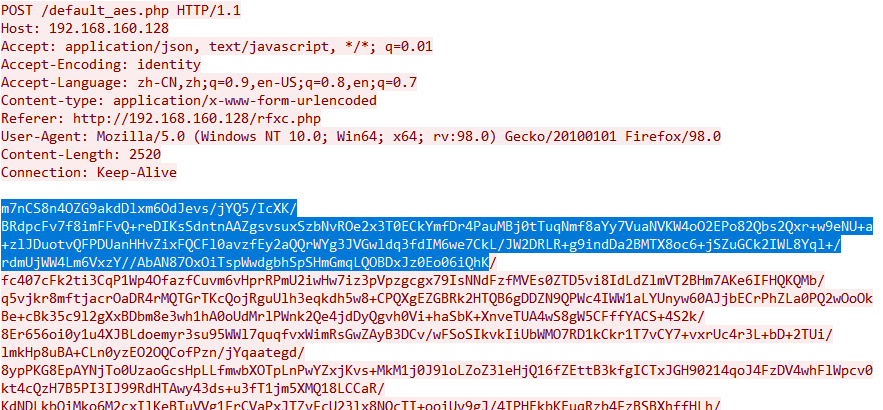
每种模式下的特征

default\_aes 在默认情况下的aes协议模式下,发送的请求包中都包含有

m7nCS8n4OZG9akdDlxm6OdJevs/jYQ5/IcXK

BRdpcFv7f8imFFvQ+reDIKsSdntnAAZgsvsuxSzbNvROe2x3T0ECkYmfDr4PauMBj0tTuqNmf8aYy7VuaNVKW4oO2EPo82Qbs2Qxr+w9eNU+a+zlJDuotvQFPDUanHHvZixFQCFl0avzfEy2aQQrWYg3JVGwldq3fdIM6we7CkL/JW2DRLR+g9indDa2BMTX8oc6+jSZuGCk2IWL8Yql+/rdmUjWW4Lm6VxzY//AbAN87OxOiTspWwdgbhSpSHmGmqLQOBDxJz0Eo06iQhK

这串加密字符,可以初步看作是默认情况下的特征,但不一定稳定

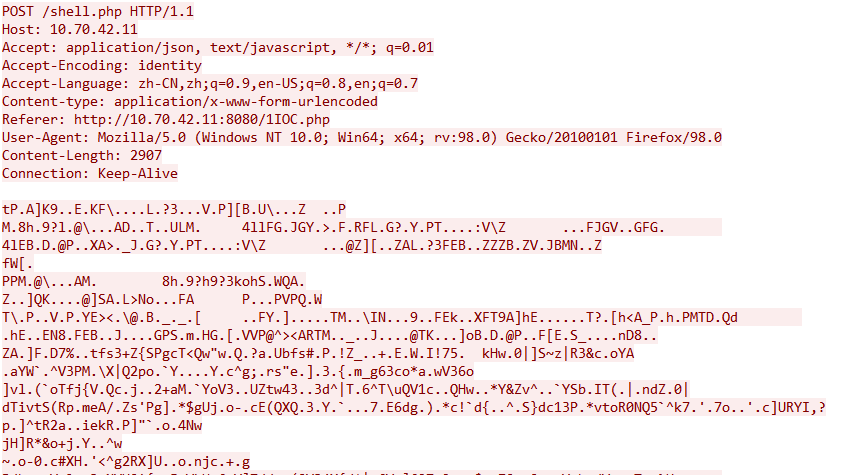




default\_xor在默认情况下, 发送的请求包中都包含有

tP.A]K9..E.KF\....L.?3...V.P][B.U\...Z ..PM.8h.9?l.@\...AD..T..ULM. 4llFG.JGY.>.F.RFL.G?.Y.PT....:V\Z ...FJGV..GFG. 4lEB.D.@P..XA>.\_J.G?.Y.PT....:V\Z ...@Z][..ZAL.?3FEB..ZZZB.ZV.JBMN..ZfW[.PPM.@\...AM. 8h.9?h9?3kohS.WQA.Z..]QK....@]SA.L>No...FA P...PVPQ.WT\.P..V.P.YE><.\@.B.\_.\_.[ ..FY.].....TM..\IN...9..FEk..XFT9A]hE......T?.[h<A\_P.h.PMTD.Qd .hE..EN8.FEB..J....GPS.m.HG.[.VVP@^><ARTM..\_..J....@TK...]oB.D.@P..F[E.S\_....nD8..

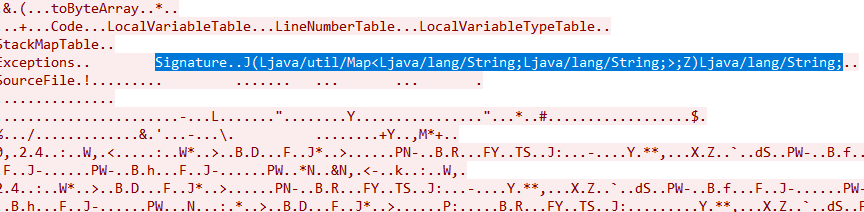
可以初步看作是默认情况下的特征





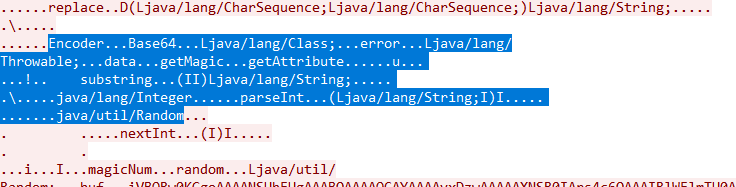
default\_image 在默认情况下, 发送的请求包中有

对java接口Signature..J(Ljava/util/Map<Ljava/lang/String;Ljava/lang/String;>;Z)Ljava/lang/String;等等这种比较明显的接口





除此之外还有明显的Encoder Base64等字符串



default\_json在默认情况下, 发送的请求包中有

{"id":"1","body":{"user":"yv66vgAAADIBMwEA ，可以初步看作是默认情况下的特征,但不一定稳定



