BurpCrypto: 万能网站密码爆破测试工具

系统安全运维 2023-04-14 08:08 本文共 4259 字 阅读完需 17 分钟

BurpCrypto是一款支持多种加密算法、或直接执行浏览器JS代码的BurpSuit插件。

一、编译

```
mvn package
```

二、为什么解决了痛点

目前越来越多的网站系统在登录接口、数据请求接口中加入各式各样的加密算法，甚至有些网站在每次请求前都动态请求加密密钥等措施，对接口渗透工作造成较大障碍。依赖于BurpSuite中的那些编码方式、Hash算法已经远远不够，通过BurpCrypto内置的RSA、AES、DES模块可应对较为简单的前端加密接口，较为复杂的加密算法，可使用ExecJS模块直接手动编写处理代码。同时为了降低ExecJS模块的上手难度，未来将推出远程JS模块仓库，支持远程加载已经测试通过的JS功能代码，方便直接调用。

三、未来开发计划

四、安装

BurpCrypto可从其官方Github页面进行下载已编译好的版本，或下载源代码本地编译，然后在BurpSuite的扩展列表中添加插件，等待Output中输出 `BurpCrypto loaded successfully!` 则表示插件加载成功。

五、基础加密模块使用

基础编码方式，由于不同网站开发人员的使用习惯，加密时所使用的密钥、加密后的密文会使用不同的编码方式。目前插件内密钥输入所支持的编码方式有如下三种

- Base64
- HEX
- UTF8String

密文输出所支持的编码方式有如下两种

- Base64
- HEX

ExecJS模块的输出内容由JS代码决定。

六、编码方式的辨别方法

为了照顾到对编码方式不了解的朋友，此处会简单讲解这些编码方式的辨别方法，已经了解的朋友可直接跳过参阅下一章节的具体使用讲解.

Base64编码与HEX编码常常用于编码二进制数据，UTF8String则是我们操作系统、网页中最常见的字符串的编码方式，下方是Base64、HEX、UTF8String编码的示例：

```
对字符串"test_z"进行Base64、HEX编码的结果
Base64：dGVzdF96
HEX：746573745f7a
UTF8String：test_z
```

AES和DES加密都属于对称加密算法，既加解密使用同一套密钥的加密算法，同时也是目前前端加密中较为常见的加密算法，目前插件支持的AES加密算法有：

- AES/CBC/PKCS5Padding
- AES/CBC/NoPadding

- AES/CBC/ZeroPadding
- AES/ECB/PKCS5Padding
- AES/ECB/NoPadding
- AES/ECB/ZeroPadding
- AES/OFB/PKCS5Padding
- AES/OFB/NoPadding
- AES/OFB/ZeroPadding
- AES/CFB/PKCS5Padding
- AES/CFB/NoPadding
- AES/CFB/ZeroPadding
- AES/CTR/PKCS5Padding
- AES/CTR/NoPadding
- AES/CTR/ZeroPadding

DES加密算法有：

- DES/CBC/PKCS5Padding
- DES/CBC/ZeroPadding
- DES/CBC/NoPadding
- DES/ECB/PKCS5Padding
- DES/ECB/ZeroPadding
- DES/ECB/NoPadding
- DES/OFB/PKCS5Padding
- DES/OFB/ZeroPadding
- DES/OFB/NoPadding

- DES/CFB/PKCS5Padding
- DES/CFB/ZeroPadding
- DES/CFB/NoPadding
- DESede/CBC/PKCS5Padding
- DESede/CBC/ZeroPadding
- DESede/CBC/NoPadding
- DESede/ECB/PKCS5Padding
- DESede/ECB/ZeroPadding
- DESede/ECB/NoPadding
- DESede/OFB/PKCS5Padding
- DESede/OFB/ZeroPadding
- DESede/OFB/NoPadding
- DESede/CFB/PKCS5Padding
- DESede/CFB/ZeroPadding
- DESede/CFB/NoPadding
- strEnc

  在前端JS中常常会使用PKCS7Padding，在本模块中可使用PKCS5Padding代替，不影响使用。

  DES加密中的strEnc算法是取自作者Guapo的一种3DES的模块，在少数系统中被使用，此处为了方便使用也引入了进来。

非对称加密算法

RSA算法则属于非对称加密算法，密钥分为公钥与私钥，暂时仅支持公钥加密，RSA加密支持两种公钥格式的输入，分别为

- X509

- ModulusAndExponent

X509密钥格式表现为一串由Base64编码后的字符串，常常以 `MIG` 开头。ModulusAndExponent(模数，指数)则表现为两个HEX编码的参数，Modulus是模数，常常较长，Exponent是指数，常常只有6位，以下为密钥示例：

```
X509:
MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCC0hrRIjb3noDWNtbDpANbjt5Iwu2NFeDwU16
Ec87ToqeoIm2KI+cOs81JP9aTDk/jkAlU97mN8wZkEMDr5utAZtMVht7GLX33Wx9XjqxUsDfsGk
qNL8dXJklWDu9Zh80Ui2Ug+340d5dZtKtd+nv09QZqGjdnSp9PTfFDBY133QIDAQAB
ModulusAndExponent:
  Modulus:
A1E4D93618B8B240530853E87738403851E15BBB77421F9B2377FB0B4F1C6FC235EAEC92EA2
5BB76AC221DCE90173A2E232FE1511909C76B15251D4059B288E709C1EF86BCF692757AAD73
6882DD1E98BEDFED9311A3C22C40657C9A52880BDC4B9E539041D44D52CB26AD13AB086F7DC
294D144D6633A62EF91CA1775EB9A09
  Exponent: 010001
```

七、使用

使用方式也较为简单，首先判断相关接口的加密算法，填入相应算法的加密密钥，点击 `Add processor` ，在弹出的加密配置命名输入对话框中，给予一个易于分辨的名称，提示 `Apply processor success!` 即表示添加成功。

此处以AES的CBC模式，填充Pkcs7，Key：Y3MxMTg1MzUyOS4x，IV：9875643210132456，Base64编码的方式做为示例。

若要删除processor则要点击 `Remove processor` ，输入刚刚编写的配置名，即可删除。

关于前端加密的分析过程可参阅jsEncrypter开发者c0ny1的文章：https://gv7.me/articles/2018/fast-locate-the-front-end-encryption-method/

八、补充阅读

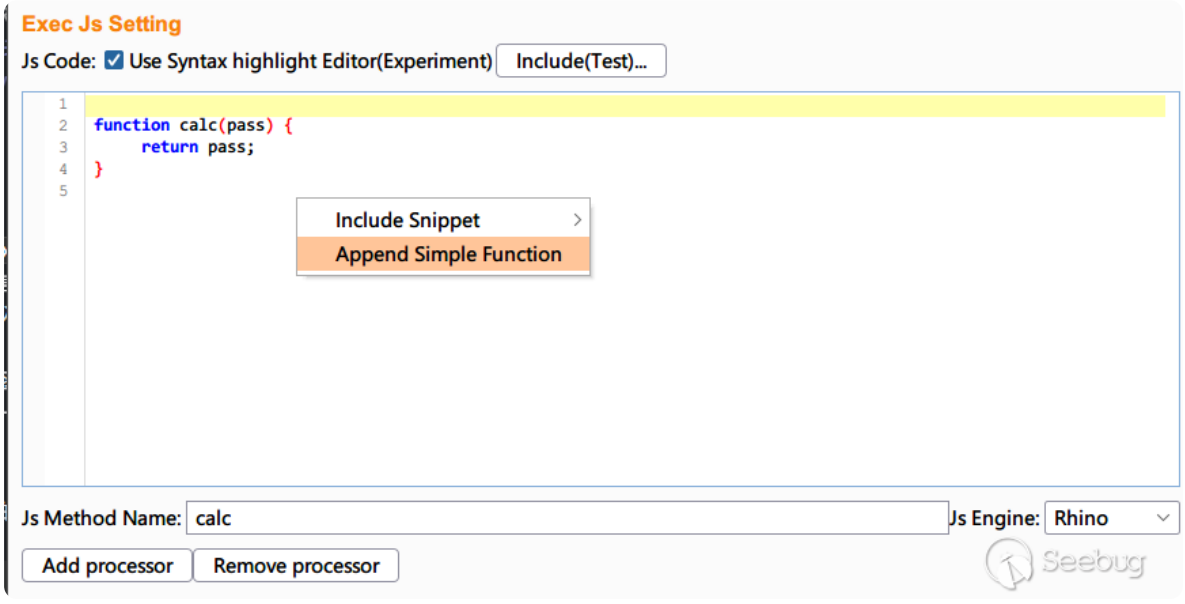
对单加密参数的登录接口进行密码爆破的一种方法，ExecJS模块使用由于软件开发的复杂性，诸如多层嵌套加密、引入时间变量、动态密钥、魔改加密算法、新算法的涌现，插件注定永远无法做到对所有加密算法的百分百覆盖，所以提供了ExecJS模块，为动手能力较强的使用者提供一条新途径。

因JS新特性的快速迭代，插件中内置了Rhino、HtmlUnit、Jre内置三种JS执行引擎，各种执行引擎的优劣势可参阅BurpCrypto未来开发计划中对于各个引擎的特性介绍。

九、编写简单的JS脚本

使用ExecJS模块前需要先切换至插件的 `ExecJS` 选项卡，像常见编程语言一样，你需要编写一个入口函数。不过不同于其他编程语言的入口函数，插件将会把待处理/加密的内容传递给入口函数的第一个参数，而你编写的入口函数则需要在处理结束后返回处理结果。

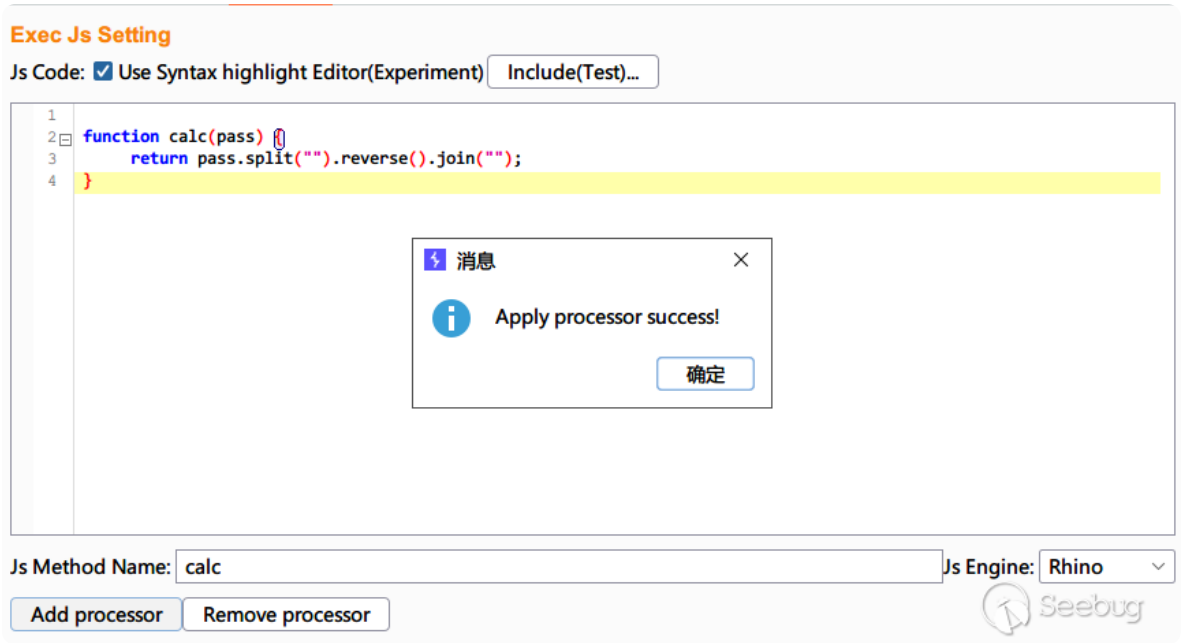
为了方便使用，插件内置了一个快速生成函数体的菜单，在代码编辑器中的右键菜单中点击 `Append Simple Function` ，即可生成一个空函数 `calc` ，并自动在下面的入口函数名填写入口函数为 `calc` 。



我们可以对该函数进行一些简单的修改，下面是一个示例脚本，该脚本将会把输入的内容倒转后再返回。

```
function calc(pass) {
    return pass.split("").reverse().join("");
}
```

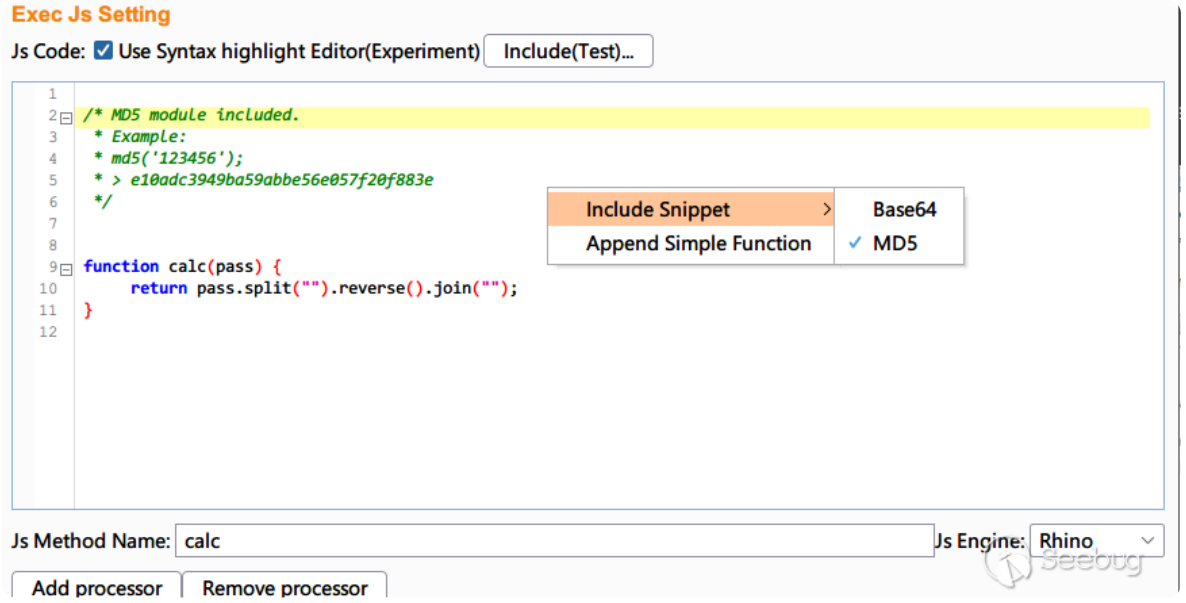
编辑完成后，点击 `Add processor` 即可添加成功。



十、引用内置JS库

目前内置的JS库只有MD5与Base64，后续版本将会上线在线JS仓库，操作步骤将会发生变动。

为避免常见库的频繁整理导入的工作量，插件目前内置了MD5和Base64库，使用方法为在编辑器的右键菜单中的 `Include Snippet` 选中需要的JS库，即可引入。

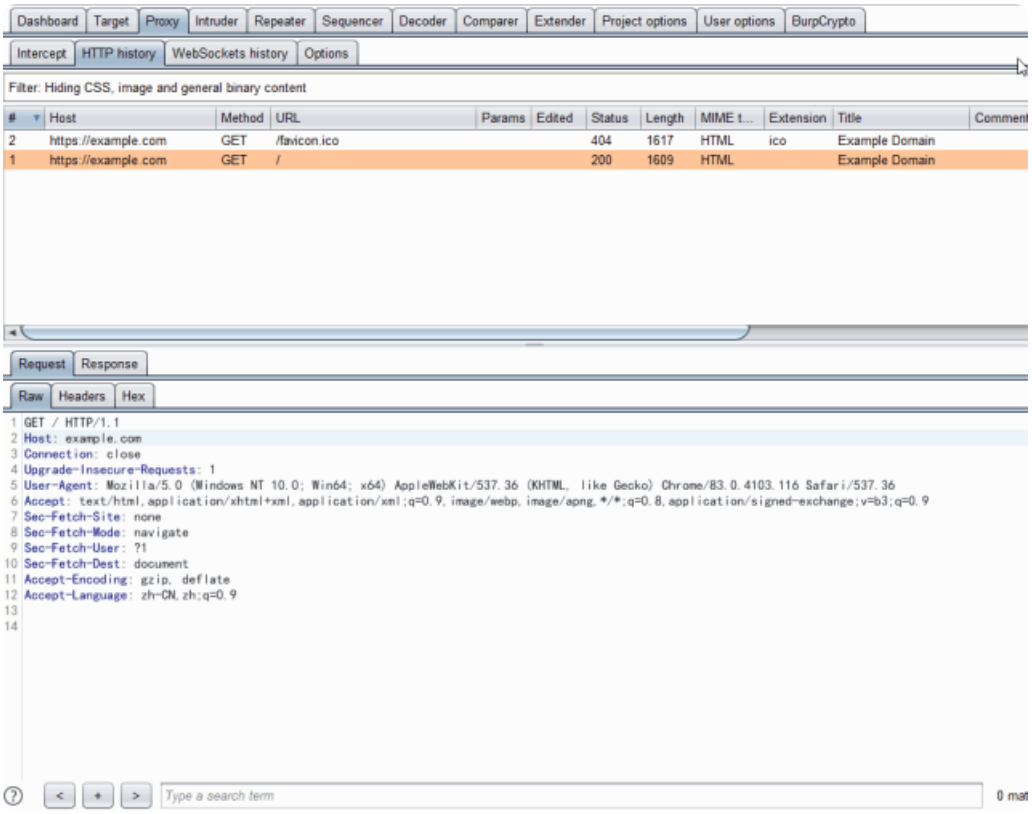


十一、在功能区中调用插件

加密，通过在上述几个模块中成功添加processor后，即可通过以下两个渠道进行使用。

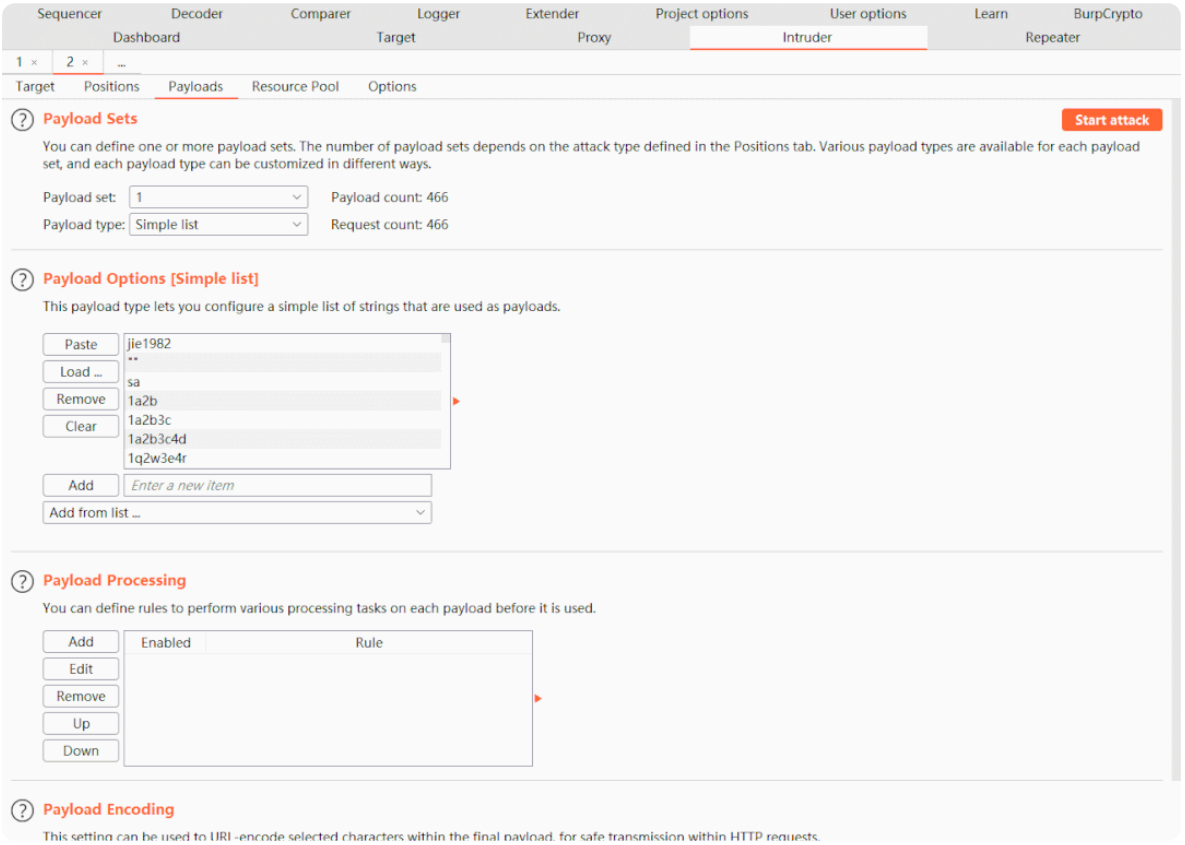
QuickCrypto（全局调用）

BurpCrypto几乎可以在BurpSuite的任何位置进行调用，调用方法也较为简单，以下动图为示例：



十二、Intruder（爆破模块）

在爆破模块中引用：

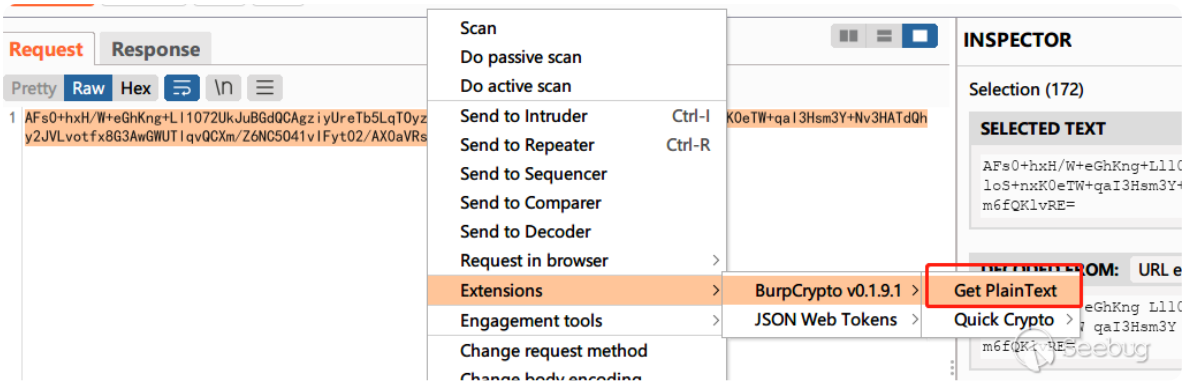


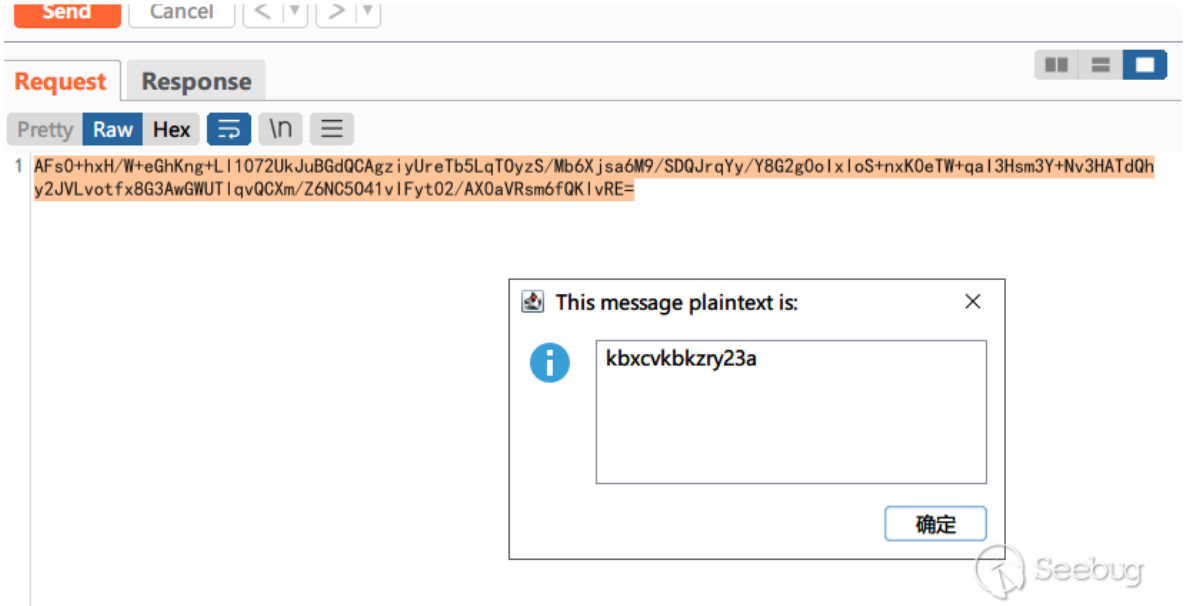
十三、解密（查询原文）

由于部分算法产生的结果具有不可逆性（哈希算法、自定义的JS代码等），所以通过本插件生成的所有结果均被保存到本地的K/V数据库中，可以通过插件中的 `Get PlainText` 功能获取原始明文。

使用方法为完整选中密文内容，右单击后找到BurpCrypto菜单中的 `Get PlainText` 功能，

此处演示的是RSA公钥加密后通过密文查询原始明文。





内容来源：https://github.com/whwlsfb/BurpCrypto

十四、结尾

以上为本插件的使用说明，后续将会将会陆续加入国密算法、ExecJS远程模块，模块互调等小功能更新与Bug修复，如果各位师傅有更多的建议也欢迎提PR或者Issure，谢谢！

如有侵权，请联系删除

好文推荐

红队打点评估工具推荐

干货|红队项目日常渗透笔记

实战|后台getshell+提权一把梭

一款漏洞查找器（挖漏洞的有力工具）

神兵利器 | 附下载 · 红队信息搜集扫描打点利器

神兵利器 | 分享 直接上手就用的内存马（附下载）

推荐一款自动向hackerone发送漏洞报告的扫描器

欢迎关注 系统安全运维

公众号