shiro

目前收集到有如下中情况： 不同语言对 base64 解码时的处理有所不同： 字符串中包含 . % ^等符号时，是选择忽略这些符号，还是报错 字符串中包含 = 符号，解析到=时，是认为解析完成了，还是忽略"等号"继续解析 Dwedwedewytrv 利用这一特性，可以混淆 rememberMe 字段后的数据：

`~!#$%^&*-_|.可以使用，但是每次只能选择一个，只能出现一次

Base64 混淆+无用数据填充

利用=号后不再解析的特性(识别到等号，不再解码=后面的内容)

利用链&内存马

利用链

利用链是目标存在哪些依赖或者说使用了哪些存在的方法，可以帮助你去一步步寻找执行点，完成漏洞的利用

```
PriorityQueue.readObject
    -> PriorityQueue.heapify()
    -> PriorityQueue.siftDown()
    -> PriorityQueue.siftDownUsingComparator()
        -> TransformingComparator.compare()
            -> InvokerTransformer.transform()
                -> TemplatesImpl.newTransformer()
                ... templates Gadgets ...
                    -> Runtime.getRuntime().exec()
```

https://www.anquanke.com/post/id/192619

内存马

通过写入内存的方式加载的木马，不写入文件，可进行直接连接。但是要明白只是写入了内存，所以重启服务，或关机以后则会断开
https://www.freebuf.com/articles/web/274466.html

key的收集

通过在github或其他网站官网上寻找使用了shiro组件的框架源码，所携带的默认的key

java反序列化的构建

当被反序列化的数据流用户可控时，那么攻击者即可通过构造恶意输入，让反序列化产生非预期的对象，在此过程中执行构造的任意代码
关键点在于用户自定义类中的 readObject() 方法形成了不安全的类，导致了反序列化安全问题
https://www.cnblogs.com/wjrblogs/p/14057785.html