Spring Data Redis <=2.1.0反序列化漏洞

rebeyond / 2018-05-15 14:14:41 / 浏览数 3843 安全技术 漏洞分析 顶(0) 踩(0)

简介

Spring Data Redis隶属于Spring Data家族, 提供简单易用的方式来访问Redis缓存。

Spring Data Redis（包括最新版本）在往Redis里面写数据的时候，默认会先对数据进行序列化，然后把序列化之后的字节码写入Redis；然后当Spring Data Redis从Redis里取数据的时候，会取出字节码进行反序列化操作，在反序列化的过程中没有对目标类进行校验和过滤，可导致远程任意代码执行。

攻击路径

1.首先准备反序列化payload，参考ysoserial系列。

2.把生成的payload写入Redis中：

redis.set("\xac\xed\x00\x05t\x00\brebeyond",Payload)

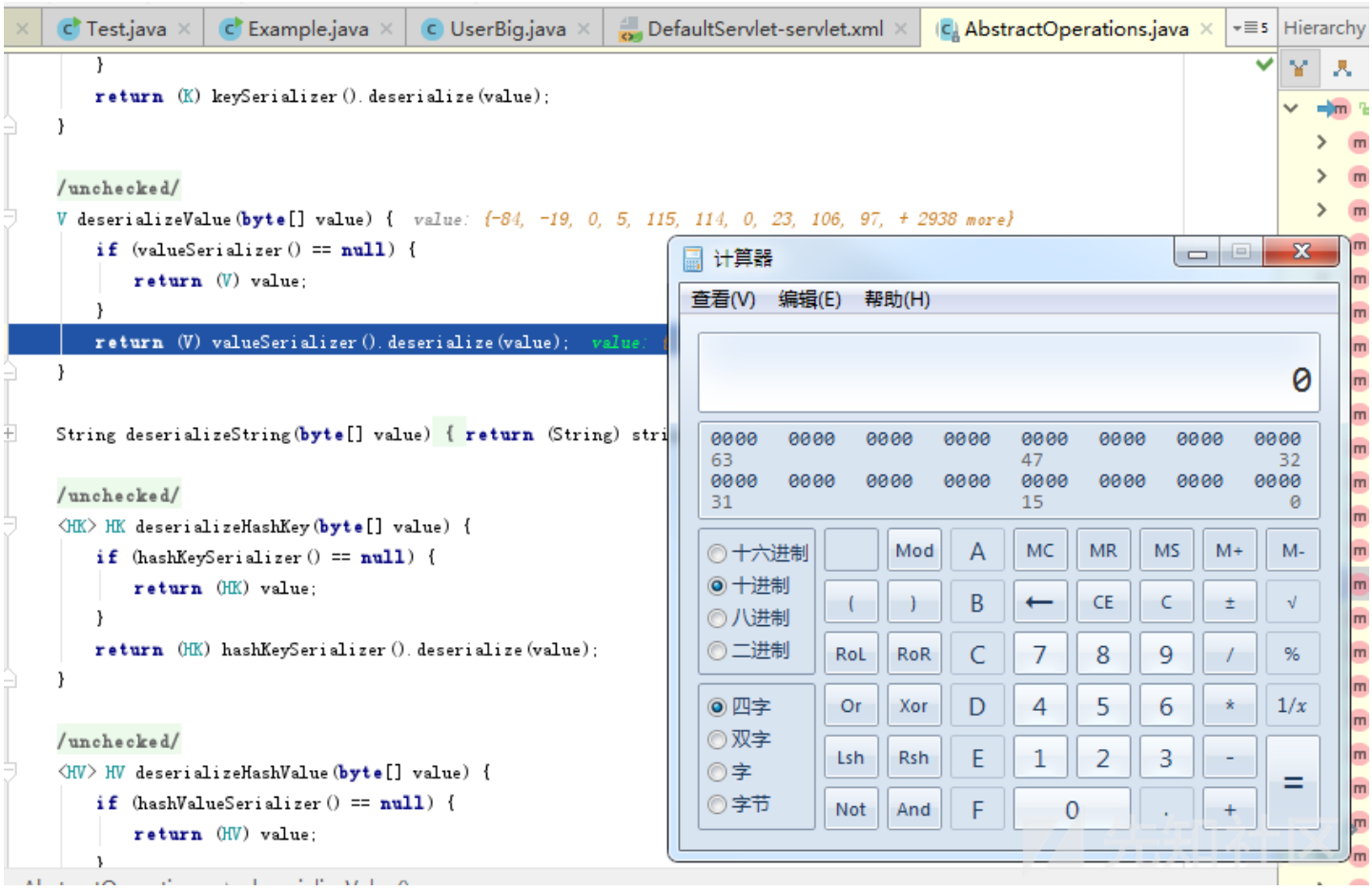
这样，名为rebeyond的key中就有了我们构造的payload。

最好选择Redis中已经存在的key，这样等Spring取数据的时候就可以触发代码执行。

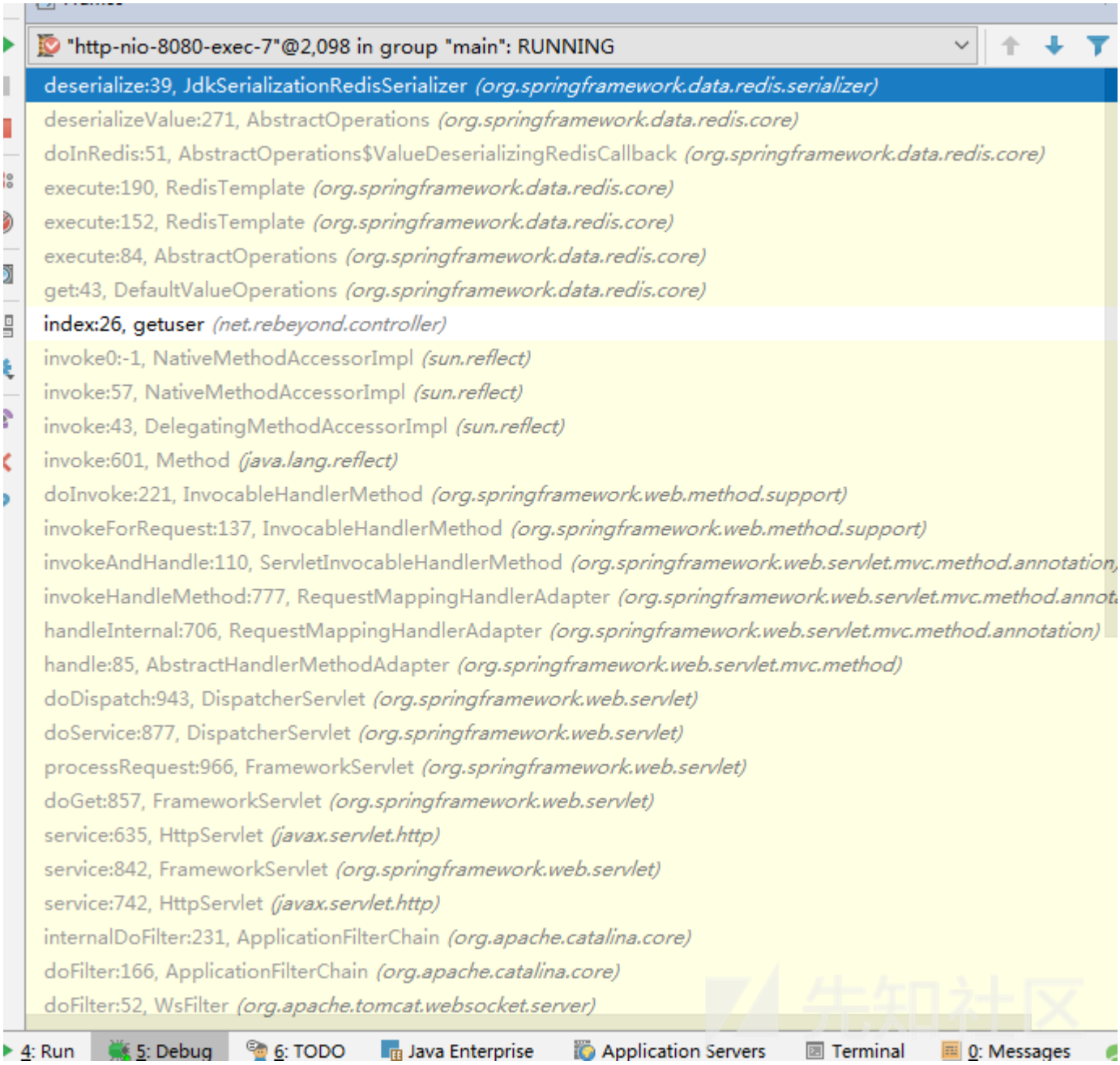
3.等待Spring读取我们已经覆盖的key，示例如下：

```
import java.io.Serializable;

@Controller
@RequestMapping("/demo")
public class getuser {

    @Autowired
    protected RedisTemplate<Serializable, Serializable> redisTemplate;

    @RequestMapping("/index")
    public String index(){

        Object result=redisTemplate.opsForValue().get("rebeyond");
        System.out.println("result:"+result);
        return "demo";
    }
}
```


4.Spring侧的机器成功弹出计算器：


Test.java
Example.java
UserBig.java
DefaultServlet-servlet.xml
AbstractOperations.java
Hierarchy
return (K) keySerializer().deserialize(value);
/unchecked/
V deserializeValue(byte[] value) { value: {-84, -19, 0, 5, 115, 114, 0, 23, 106, 97, + 2938 more}
if (valueSerializer() == null) {
return (V) value;
return (V) valueSerializer().deserialize(value);
String deserializeString(byte[] value) { return (String) stri
/unchecked/
<HK> HK deserializeHashKey(byte[] value) {
if (hashKeySerializer() == null) {
return (HK) value;
return (HK) hashKeySerializer().deserialize(value);
/unchecked/
<HV> HV deserializeHashValue(byte[] value) {
if (hashValueSerializer() == null) {
return (HV) value;
计算器
查看(V) 编辑(E) 帮助(H)
0
63 47 32
31 15 0
十六进制
十进制
八进制
二进制
四字
双字
字
字节
Mod A MC MR MS M+ M-
( ) B ← CE C ± √
RoL RoR C 7 8 9 / %
Or Xor D 4 5 6 * 1/x
Lsh Rsh E 1 2 3 - =
Not And F 0 . +

调用栈如下：



其他

This vulnerability is tested on jdk1.8.0_144+spring 5.0.3+spring data redis 2.0.3+ commons-collections4:4.0

Common-collections4 is not necessary.Some modifications to payload can be applied to <=jdk8u20 without Common-collections4.

尝试把这个问题提给Spring，不过Spring认为Redis在内网，开发人员有必要保证Redis的安全，所以没有认可该问题，只是更新了他们的产品guidelines
，让用户确保自己的Redis用在安全网络中。下图是pivotal的答复。认为Redis在内网就是安全的有点太乐观，我觉得对于spring来讲更好的解决方案是把默认的序列化引擎

> Hello qiuyongyong,
>
> We updated our guidelines regarding serializer use accordingly and consider this ticket resolved. See [0] and [1] for further reference.
>
> [0] https://jira.spring.io/browse/DATAREDIS-780
> [1] https://github.com/spring-projects/spring-data-redis/commit/1f6790b10099f26c23c46ae5a099ba1023f055b1
>
> Thanks,
> Brian Dussault
> Pivotal Security Team

[0] https://jira.spring.io/browse/DATAREDIS-780

[1] https://github.com/spring-projects/spring-data-redis/commit/1f6790b10099f26c23c46ae5a099ba1023f055b1

点击收藏 | 2 关注 | 2

上一篇：JSON Web Token (J... 下一篇：ZipperDown漏洞简单分析及防护

1. 0 条回复
   - 动动手指，沙发就是你的了！

登录 后跟帖

先知社区

现在登录

热门节点

技术文章

社区小黑板

目录