MySQL JDBC中detectCustomCollations触发点不同版本的差异

nbxiglk (/u/238) / 2022-03-03 23:29:05 / 浏览数 43

JDBC Mysql Attack

关于JDBC Attack中对Mysql的利用,主要是通过两种方式触发反序列化,一是通过queryInterceptors，二是通过detectCustomCollations，原理细节不再赘述,具体可先学习参考fnmsd师傅这篇参考(http://https://www.anquanke.com/post/id/203086#h2-4)。

关于detectCustomCollations

在上面参考文章中,写到detectCustomCollations触发的方法不适用于5.1.41及以上和5.1.18以下，因为其不再使用getObject的方式获取SHOW COLLATION的结果，导致此方法失效,但在学习的过程中发现与原文章有一些差异,故此总结。

<=5.1.18

在5.1.18及以下版本中对 `SHOW COLLATION` 返回的结果并未使用getObject(),导致无法触发反序列化。

```
if (sortedCollationMap == null) {
    sortedCollationMap = new TreeMap();
    stmt = this.getMetadataSafeStatement();
    results = stmt.executeQuery( sql: "SHOW COLLATION");

    while(results.next()) {
        String charsetName = results.getString( columnIndex: 2);
        Integer charsetIndex = results.getInt( columnIndex: 3);
        sortedCollationMap.put(charsetIndex, charsetName);
```

(https://xzfile.aliyuncs.com/media/upload/picture/20220222162605-142e1f72-93b9-1.png)

5.1.19-5.1.39

而在5.1.19-5.1.39版本之间,对 `SHOW COLLATION` 返回的结果处理逻辑如下

```
if (this.versionMeetsMinimum( major: 4, minor: 1, subminor: 0) && this.getDetectCustomCollations()) {
    java.sql.Statement stmt = null;
    ResultSet results = null;

    try {
        sortedCollationMap = new TreeMap();
        customCharset = new HashMap();
        customMblen = new HashMap();
        stmt = this.getMetadataSafeStatement();

        try {
            results = stmt.executeQuery( sql: "SHOW COLLATION");
            if (this.versionMeetsMinimum( major: 5, minor: 0, subminor: 0)) {
                Util.resultSetToMap(sortedCollationMap, results, key: 3, value: 2);
```

(https://xzfile.aliyuncs.com/media/upload/picture/20220222162904-7e91f848-93b9-1.png)
会对SHOW COLLATION返回结果调用 `Util.resultSetToMap` 进行处理，而 `resultSetToMap` 则和 `ServerStatusDiffInterceptor` 相同,最后走到 `ResultSetImpl#getObject` 触发反序列化。
流程如下:

```
readObject:1397, HashMap (java.util)
invoke0:-1, NativeMethodAccessorImpl (sun.reflect)
invoke:62, NativeMethodAccessorImpl (sun.reflect)
invoke:43, DelegatingMethodAccessorImpl (sun.reflect)
invoke:498, Method (java.lang.reflect)
invokeReadObject:1058, ObjectStreamClass (java.io)
readSerialData:1900, ObjectInputStream (java.io)
readOrdinaryObject:1801, ObjectInputStream (java.io)
readObject0:1351, ObjectInputStream (java.io)
readObject:371, ObjectInputStream (java.io)
getObject:5016, ResultSetImpl (com.mysql.jdbc)
resultSetToMap:508, Util (com.mysql.jdbc)
buildCollationMapping:1004, ConnectionImpl (com.mysql.jdbc)
```

(https://xzfile.aliyuncs.com/media/upload/picture/20220222163359-2e513276-93ba-1.png)

5.1.40-5.1.48

而从5.1.40开始处理代码逻辑如下

```
try {
    results = stmt.executeQuery( sql: "SHOW COLLATION");

    while(results.next()) {
        int collationIndex = ((Number)results.getObject( columnIndex: 3)).intValue();
        String charsetName = results.getString( columnIndex: 2);
        if (collationIndex > 2048 || !charsetName.equals(CharsetMapping.getMysqlCharsetNameForCollationIn
            ((Map)customCharset).put(collationIndex, charsetName);
        }
```

(https://xzfile.aliyuncs.com/media/upload/picture/20220222163502-544ffa0c-93ba-1.png)

确实不再直接调用 `ResultSetImpl#getObject` ,但是对 `SHOW COLLATION` 结果的第三列直接调用了results.getObject(),最后还是进入的ResultSetImpl#getObject中.

其中满足字段类型为-4,-3,-2(blob,bit,binary)时会进入 `getObjectDeserializingIfNeeded` 方法,直接用恶意mysql服务器(https://github.com/fnmsd/MySQL_Fake_Server)的设置,修改一下代码将第三个字段也填充为序列化数据即可.

```
        }
    case -4:
    case -3:
    case -2:
        if (field.getMysqlType() == 255) {
            return this.getBytes(columnIndex);
        }

        return this.getObjectDeserializingIfNeeded(columnIndex);
    case -1:
```

(https://xzfile.aliyuncs.com/media/upload/picture/20220222163559-75c6b824-93ba-1.png)

该方法和之前类似进行了反序列化操作,其中通过 `this.connection.getAutoDeserialize()` 来确定是否进行反序列化,在url中同之前一样设置 `autoDeserialize=true` 即可.

```
private Object getObjectDeserializingIfNeeded(int columnIndex) throws SQLException {
    Field field = this.fields[columnIndex - 1];
    if (!field.isBinary() && !field.isBlob()) {
        return this.getBytes(columnIndex);
    } else {
        byte[] data = this.getBytes(columnIndex);
        if (!this.connection.getAutoDeserialize()) {
            return data;
        } else {
            Object obj = data;
            if (data != null && data.length >= 2) {
                if (data[0] != -84 || data[1] != -19) {
                    return this.getString(columnIndex);
                }

                try {
                    ByteArrayInputStream bytesIn = new ByteArrayInputStream(data);
                    ObjectInputStream objIn = new ObjectInputStream(bytesIn);
                    obj = objIn.readObject();
                    objIn.close();
                    bytesIn.close();
```

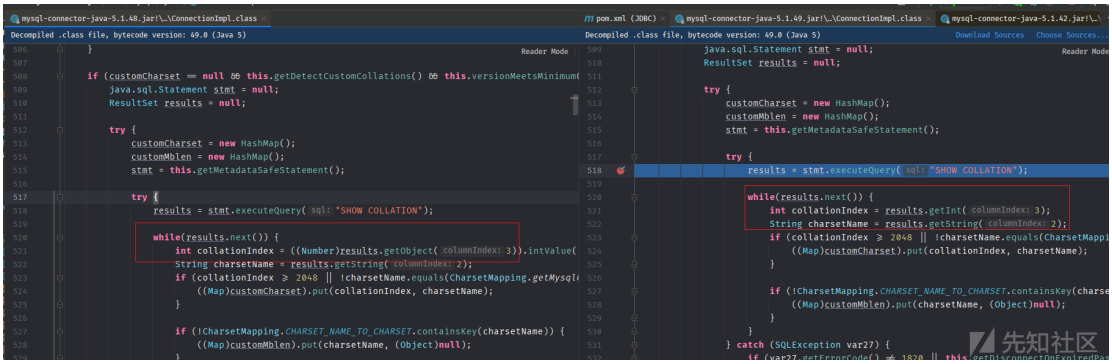
(https://xzfile.aliyuncs.com/media/upload/picture/20220222163644-90e45706-93ba-1.png)

```
readObject:1397, HashMap (java.util)
invoke0:-1, NativeMethodAccessorImpl (sun.reflect)
invoke:62, NativeMethodAccessorImpl (sun.reflect)
invoke:43, DelegatingMethodAccessorImpl (sun.reflect)
invoke:498, Method (java.lang.reflect)
invokeReadObject:1058, ObjectStreamClass (java.io)
readSerialData:1900, ObjectInputStream (java.io)
readOrdinaryObject:1801, ObjectInputStream (java.io)
readObject0:1351, ObjectInputStream (java.io)
readObject:371, ObjectInputStream (java.io)
getObjectDeserializingIfNeeded:4571, ResultSetImpl (com.mysql.jdbc)
getObject:4536, ResultSetImpl (com.mysql.jdbc)
buildCollationMapping:924, ConnectionImpl (com.mysql.jdbc)
```

(https://xzfile.aliyuncs.com/media/upload/picture/20220222163651-94ef0878-93ba-1.png)

5.1.49

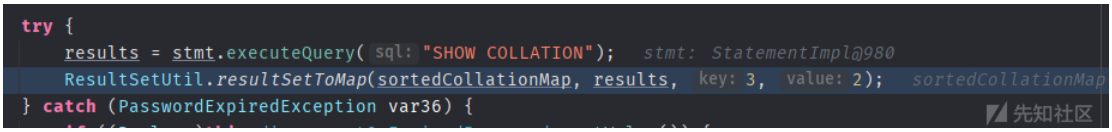
而直到5.1.49才真正没有调用getObject().



(https://xzfile.aliyuncs.com/media/upload/picture/20220222163711-a0ac20d8-93ba-1.png)

6.0.2

但在6.0.2开始,版本中却又直接调用了 `ResultSetUtil.resultSetToMap` ， 又回到了 `ServerStatusDiffInterceptor` 那条链中,同样可以触发反序列化,实测直到6.0.6版本都是这样。

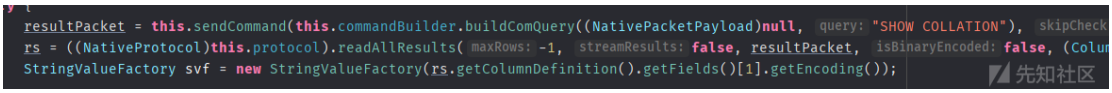


(https://xzfile.aliyuncs.com/media/upload/picture/20220222163738-b0e6b58a-93ba-1.png)

(https://xzfile.aliyuncs.com/media/upload/picture/20220222163746-b602c4be-93ba-1.png)

8.x

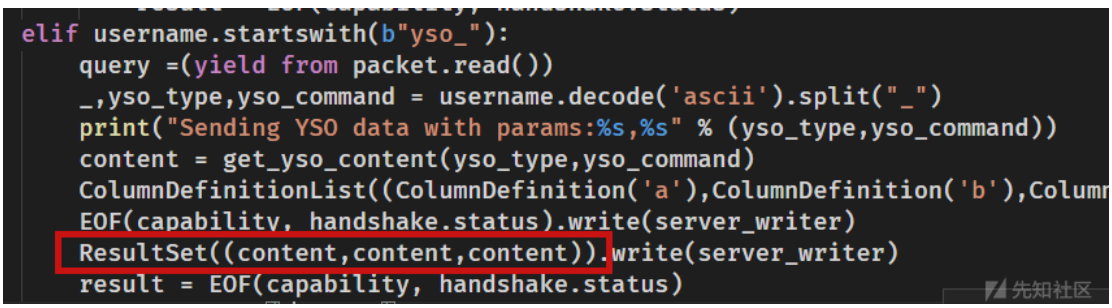
而在8.x版本获取 `SHOW COLLATION` 时又不一样了,才终于又无法利用该点触发了。



(https://xzfile.aliyuncs.com/media/upload/picture/20220222163813-c605f43a-93ba-1.png)

总结

detectCustomCollations可触发的版本:5.1.18< version <=6.0.6(5.1.49除外)。
在恶意mysql服务器 (https://github.com/fnmsd/MySQL_Fake_Server)的基础上修改一下填充第三个字段为yso序列化的数据即可通用。



(https://xzfile.aliyuncs.com/media/upload/picture/20220222164922-547a8432-93bc-1.png)