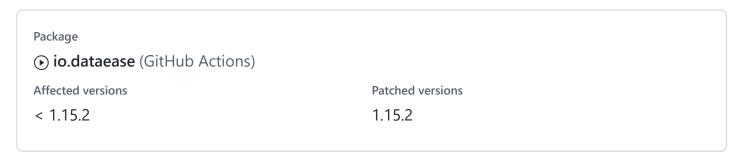
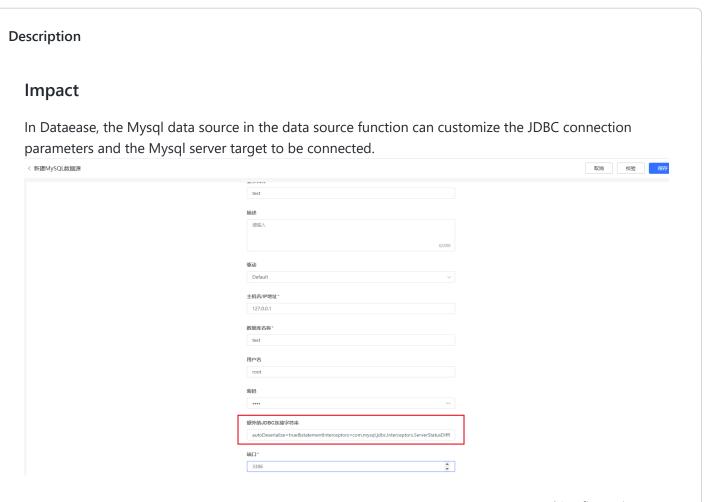


# Dataease Mysql Data Source JDBC Connection Parameters Not Verified Leads to Deserialization Vulnerability

High fit2cloudrd published GHSA-q4qq-jhjv-7rh2 on Oct 13





In backend/src/main/java/io/dataease/provider/datasource/JdbcProvider.java, MysqlConfiguration class don't filter any parameters, directly concat user input.

```
@Getter
@Setter
public class MysqlConfiguration extends JdbcConfiguration {
    private String driver = "com.mysql.jdbc.Driver";
    private String extraParams = "characterEncoding=UTF-8&connectTimeout=5000&useSSL=false&allow
    public String getJdbc() {
        if(StringUtils.isEmpty(extraParams.trim())){
            return "jdbc:mysql://HOSTNAME:PORT/DATABASE"
                    .replace("HOSTNAME", getHost().trim())
                    .replace("PORT", getPort().toString().trim())
                    .replace("DATABASE", getDataBase().trim());
        }else {
            return "jdbc:mysql://HOSTNAME:PORT/DATABASE?EXTRA PARAMS"
                    .replace("HOSTNAME", getHost().trim())
                    .replace("PORT", getPort().toString().trim())
                    .replace("DATABASE", getDataBase().trim())
                    .replace("EXTRA_PARAMS", getExtraParams().trim());
        }
    }
}
```

So, if the attack add some parameters in JDBC url, and connect to evil mysql server, he can trigger the mysql jdbc deserialization vulnerability, and eventually the attacker can execute through the deserialization vulnerability system commands and obtain server privileges.

Affected versions: < 1.15.2

### **Patches**

The vulnerability has been fixed in v1.15.2.

```
dataease/backend/src/main/java/io/dataease/dto/datasource/MysqlConfiguration.java
Line 19 in 6c3a011

19 public String getJdbc() {
```

the MysqlConfiguration class use illegalParameters filter illegal parameters to fix this vulnerability.

```
@Getter
@Setter
public class MysqlConfiguration extends JdbcConfiguration {
    private String driver = "com.mysql.jdbc.Driver";
    private String extraParams = "characterEncoding=UTF-
8&connectTimeout=5000&useSSL=false&allowPublicKeyRetrieval=true&zeroDateTimeBehavior=convertToNu
    private List<String> illegalParameters = Arrays.asList("autoDeserialize",
```

```
"queryInterceptors", "statementInterceptors", "detectCustomCollations");
    public String getJdbc() {
        if (StringUtils.isEmpty(extraParams.trim())) {
            return "jdbc:mysql://HOSTNAME:PORT/DATABASE"
                    .replace("HOSTNAME", getHost().trim())
                    .replace("PORT", getPort().toString().trim())
                    .replace("DATABASE", getDataBase().trim());
        } else {
            for (String illegalParameter : illegalParameters) {
                if (getExtraParams().contains(illegalParameter)) {
                    throw new RuntimeException("Illegal parameter: " + illegalParameter);
                }
            }
            return "jdbc:mysql://HOSTNAME:PORT/DATABASE?EXTRA_PARAMS"
                    .replace("HOSTNAME", getHost().trim())
                    .replace("PORT", getPort().toString().trim())
                    .replace("DATABASE", getDataBase().trim())
                    .replace("EXTRA_PARAMS", getExtraParams().trim());
        }
    }
}
```



# Workarounds

It is recommended to upgrade the version to v1.15.2.

## For more information

If you have any questions or comments about this advisory:

- Open an issue in https://github.com/dataease/dataease
- Email us at wei@fit2cloud.com

#### Severity



#### **CVE ID**

CVE-2022-39312

#### Weaknesses

CWE-20

# Credits

