Spring Cloud Function SPEL Expression injection vulnerability

Author:h1biki

Name_zh	Spring Cloud Function SPEL 表达式注入漏洞
Name_en	Spring Cloud Function SPEL Expression injection vulnerability
CVE	----
CVSS 评分	8.4
威胁等级	High
CNNVD	-----
其他 id	-----
受影响软件	Spring Cloud Function

简介

Spring Cloud Function 是基于 Spring Boot 的函数计算框架，它抽象出所有传输细节和基础设施，允许开发人员保留所有熟悉的工具和流程，并专注于业务逻辑。

近日，Spring Cloud Function 修复了一个 SpEL 表达式注入漏洞，可利用该漏洞在未认证的情况下在目标 Spring Cloud Function 系统中远程执行任意代码。由于 Spring Cloud Function 将请求头中的“spring.cloud.function.routing-expression”参数作为 SpEL 表达式进行处理，造成 SpEL 表达式注入漏洞，成功利用此漏洞可实现任意代码执行。目前漏洞的 PoC/EXP 已公开，建议相关用户

尽快采取安全措施。

Spring cloud function is a function calculation framework based on spring boot. It abstracts all transmission details and infrastructure, allowing developers to retain all familiar tools and processes and focus on business logic.

Recently, spring cloud function fixed a spel expression injection vulnerability, which can be used to remotely execute arbitrary code in the target spring cloud function system without authentication.

The spring cloud function processes the "spring. Cloud. Function. Routing expression" parameter in the request header as a spiel expression, resulting in a spiel expression injection vulnerability. Successful exploitation of this vulnerability can enable arbitrary code execution. At present, the POC / exp of the vulnerability has been disclosed. It is recommended that relevant users take security measures as soon as possible.

漏洞影响

3.0.0.M3 <= Spring Cloud Function <=3.2.2

漏洞复现

实验环境

准备两台虚拟机

kali.2020　　192.168.160.128

Windows 10　　10.70.42.11

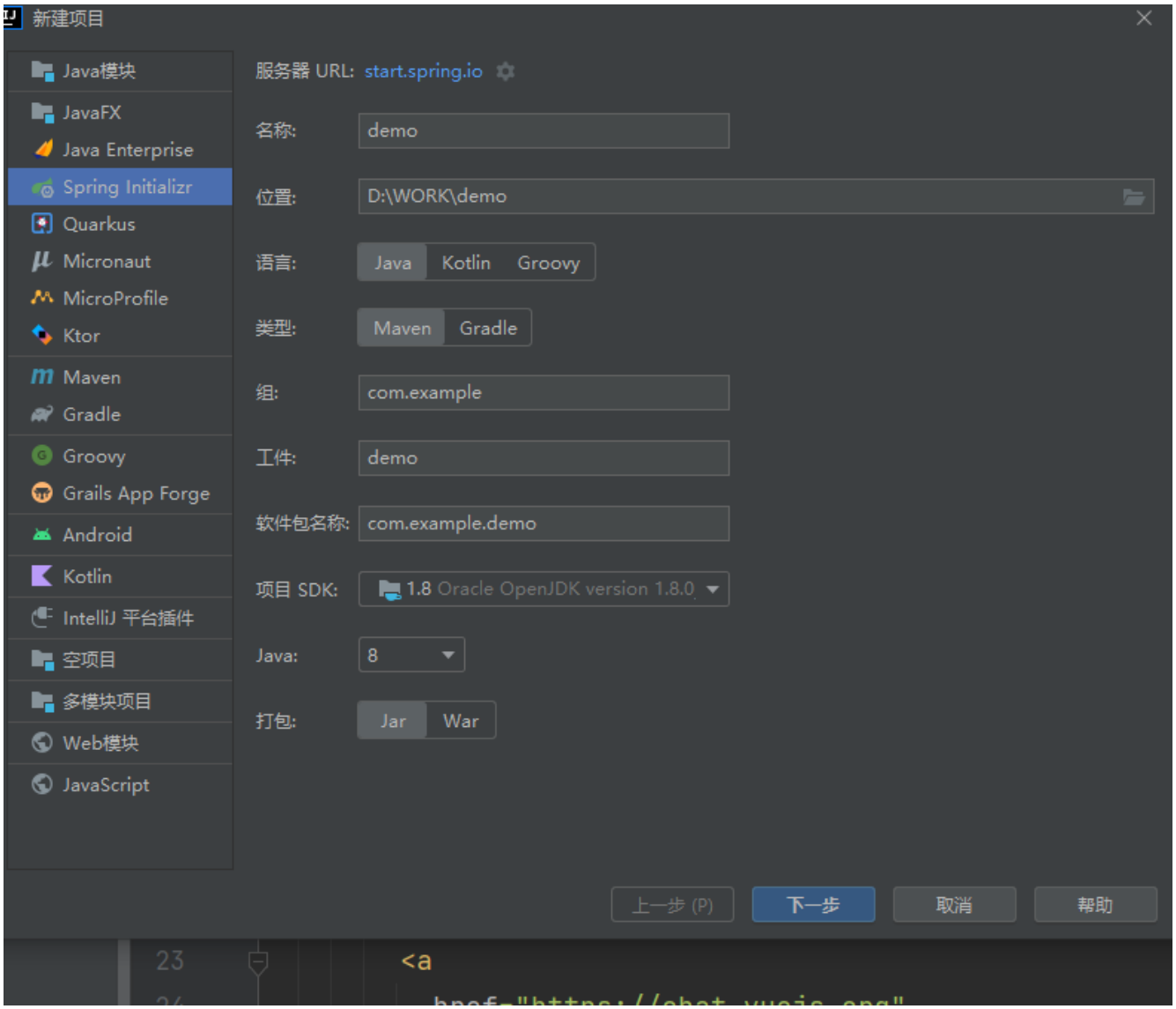
DOLIBARR：　　7.0.0

接下来利用这两台主机进行试验

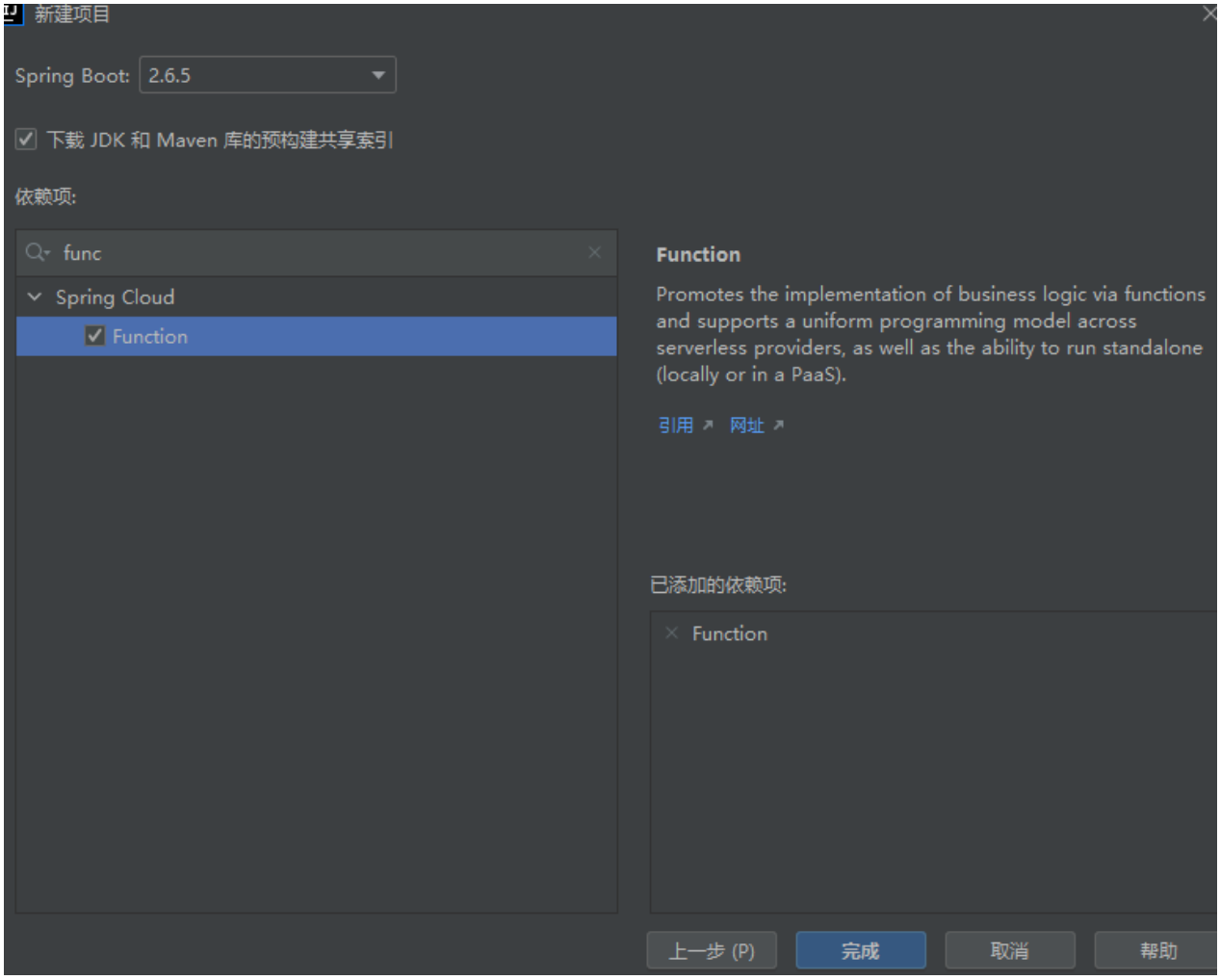
我们使用 github 上现有的项目

编译好的项目 https://github.com/Pizz33/Spring-Cloud-Function-SpEL

使用 idea 新增一个 spring Initializr 项目，选择与环境匹配的 java 版本



添加 spring web 和 function



右侧栏选择 maven-package 编译 jar 包

我们编译完成后运行 jar 包

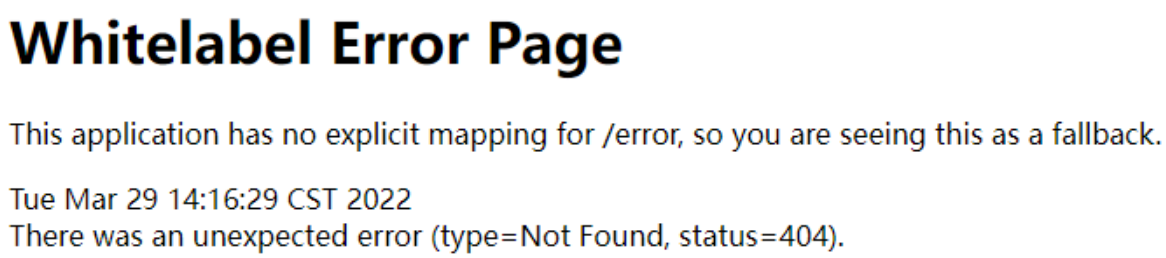
java -jar demo-0.0.1-SNAPSHOT.jar

```
PS C:\Users\nsfocus\Desktop\Spring-Cloud-Function-SpEL-main> java -jar ./demo-0.0.1-SNAPSHOT.jar

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::                (v2.6.5)

2022-03-29 14:14:58.535  INFO 10460 --- [           main] com.example.demo.DemoApplication
ers\nsfocus\Desktop\Spring-Cloud-Function-SpEL-main\demo-0.0.1-SNAPSHOT.jar started by nsfocus in
2022-03-29 14:14:58.541  INFO 10460 --- [           main] com.example.demo.DemoApplication
2022-03-29 14:15:01.234  INFO 10460 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer
2022-03-29 14:15:01.238  INFO 10460 --- [           main] o.a.catalina.core.AprLifecycleListener
2022-03-29 14:15:01.239  INFO 10460 --- [           main] o.a.catalina.core.AprLifecycleListener
rue].
2022-03-29 14:15:01.239  INFO 10460 --- [           main] o.a.catalina.core.AprLifecycleListener
2022-03-29 14:15:01.281  INFO 10460 --- [           main] o.a.catalina.core.AprLifecycleListener
2022-03-29 14:15:01.370  INFO 10460 --- [           main] o.apache.catalina.core.StandardService
2022-03-29 14:15:01.371  INFO 10460 --- [           main] org.apache.catalina.core.StandardEngine
2022-03-29 14:15:01.735  INFO 10460 --- [           main] o.a.c.c.C.[Tomcat].[localhost].[/]
2022-03-29 14:15:01.735  INFO 10460 --- [           main] w.s.c.ServletWebServerApplicationContex
2022-03-29 14:15:02.621  INFO 10460 --- [           main] o.s.c.f.web.mvc.FunctionHandlerMapping
egistry@732f29af
2022-03-29 14:15:02.743  WARN 10460 --- [           main] ConfigServletWebServerApplicationContex
ramework.context.ApplicationContextException: Failed to start bean 'webServerStartStop'; nested e
2022-03-29 14:15:02.755  INFO 10460 --- [           main] o.apache.catalina.core.StandardService
```

访问 127.0.0.1:8080 出现下图说明搭建成功



漏洞触发

可以看到官网给出的 poc 代码如下：

```
+         StepVerifier.create(resultFlux).expectError().verify();
+     }
+
+     @SuppressWarnings({ "unchecked", "rawtypes" })
+     @Test
+     public void failWithHeaderProvidedExpressionAccessingRuntime() {
+         FunctionCatalog functionCatalog = this.configureCatalog();
+         Function function = functionCatalog.lookup(RoutingFunction.FUNCTION_NAME);
+         assertThat(function).isNotNull();
+         Message<String> message = MessageBuilder.withPayload("hello")
+                 .setHeader(FunctionProperties.PREFIX + ".routing-expression",
+                         "T(java.lang.Runtime).getRuntime().exec(\"open -a calculator.app\")")
+                 .build();
+         try {
+             function.apply(message);
+             fail();
+         }
+         catch (Exception e) {
+             assertThat(e.getMessage()).isEqualTo("EL1005E: Type cannot be found
  'java.lang.Runtime'");
+         }
```

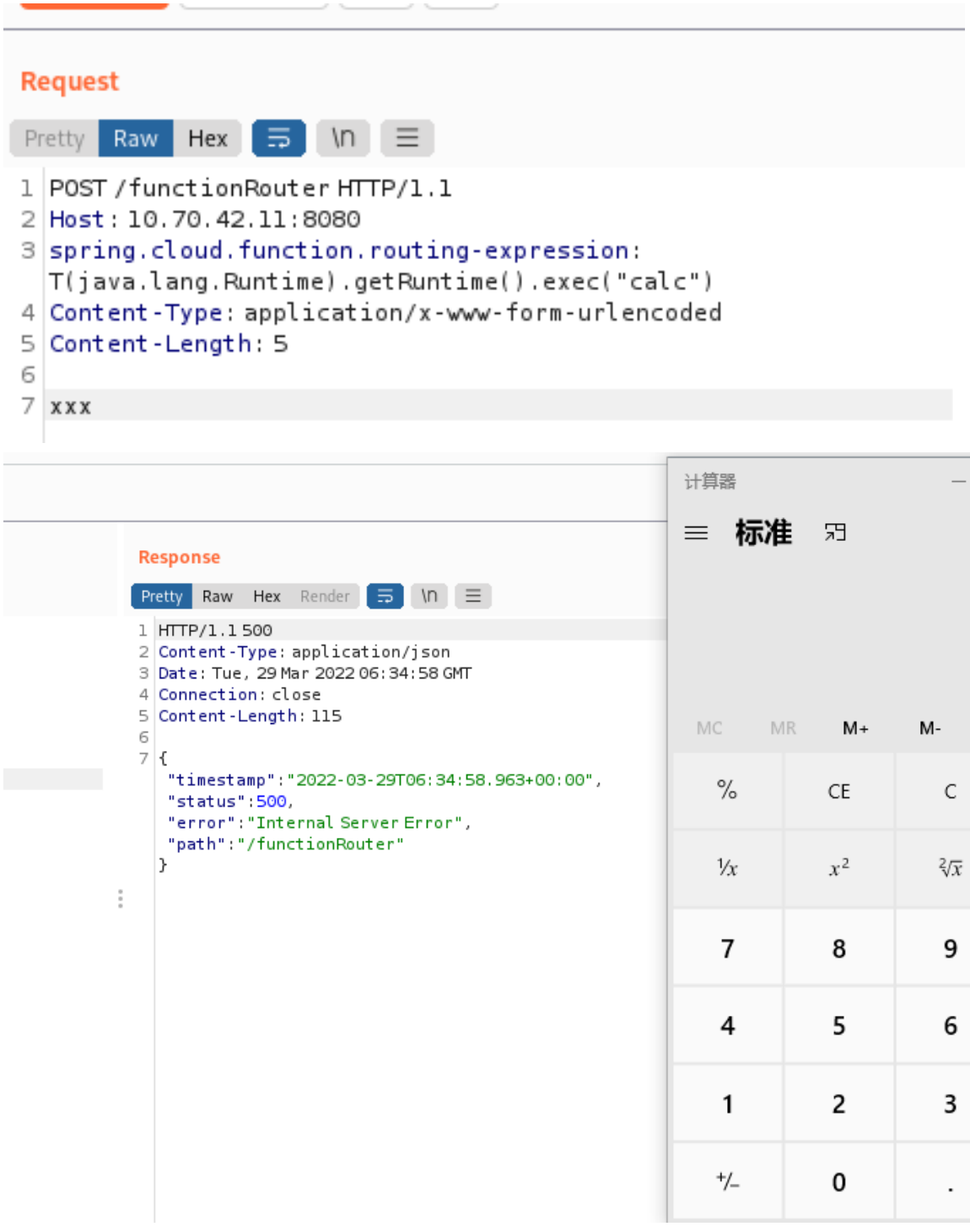
就是在请求的 headers 头上添加一个 spring.cloud.function.routing-expression

参数

SpringCloud Function 会直接将其参数内容直接带入到 SPEL 中查询，造成 SPEL

漏洞注入。

发送 poc，成功执行命令



漏洞分析

缺陷前置条件：properties 中配置 spring.cloud.function.definition=functionRouter

由于 Spring Cloud Function 中 RoutingFunction 类的 apply 方法将请求头中的

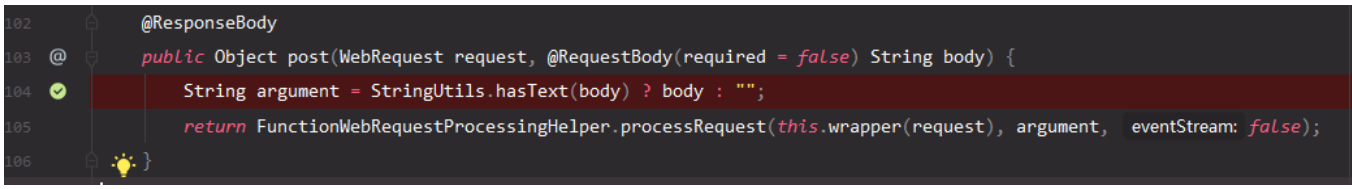
“spring.cloud.function.routing-expression”参数作为 Spel 表达式进行处理，造成了 Spel 表达式注入漏洞

RoutingFunction 其功能的目的本身就是为了微服务应运而生的，可以直接通过 HTTP 请求与单个的函数进行交互，同时为 spring.cloud.function.definition 参数提供您要调用的函数的名称。

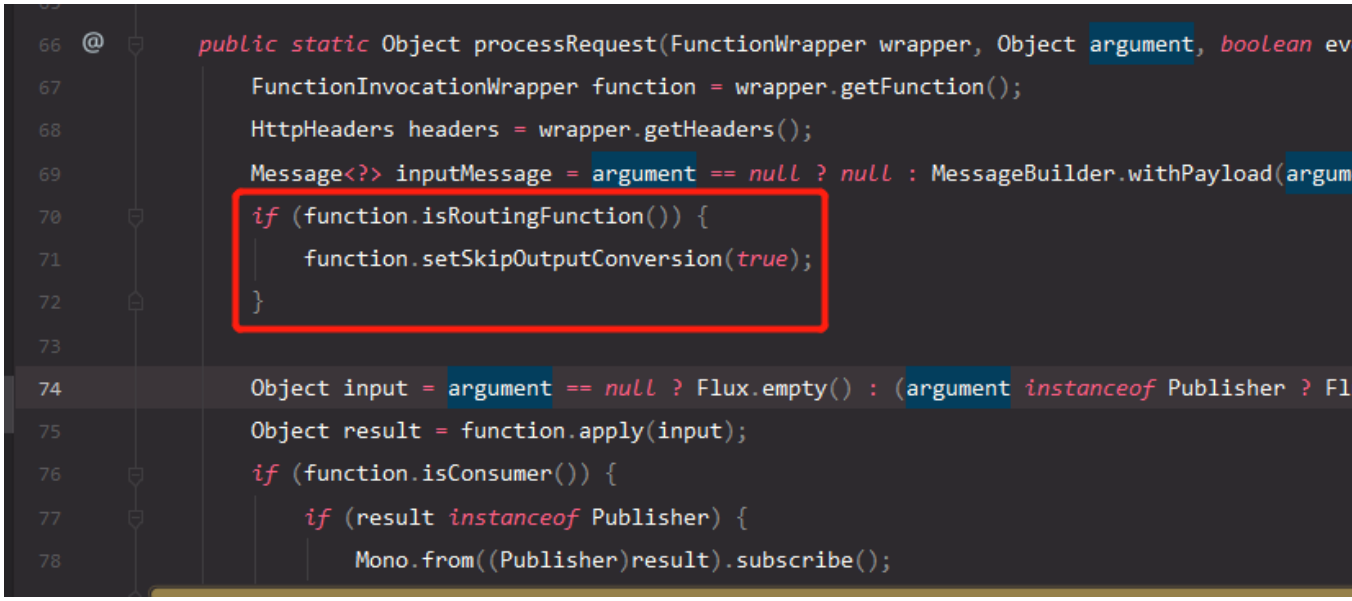
根据上述所说，漏洞是存在与 header 头的 spring.cloud.function.routing-expression 参数

我们就开始从 SpringCloud Function 的 Controller 处理来一步步往下跟入。

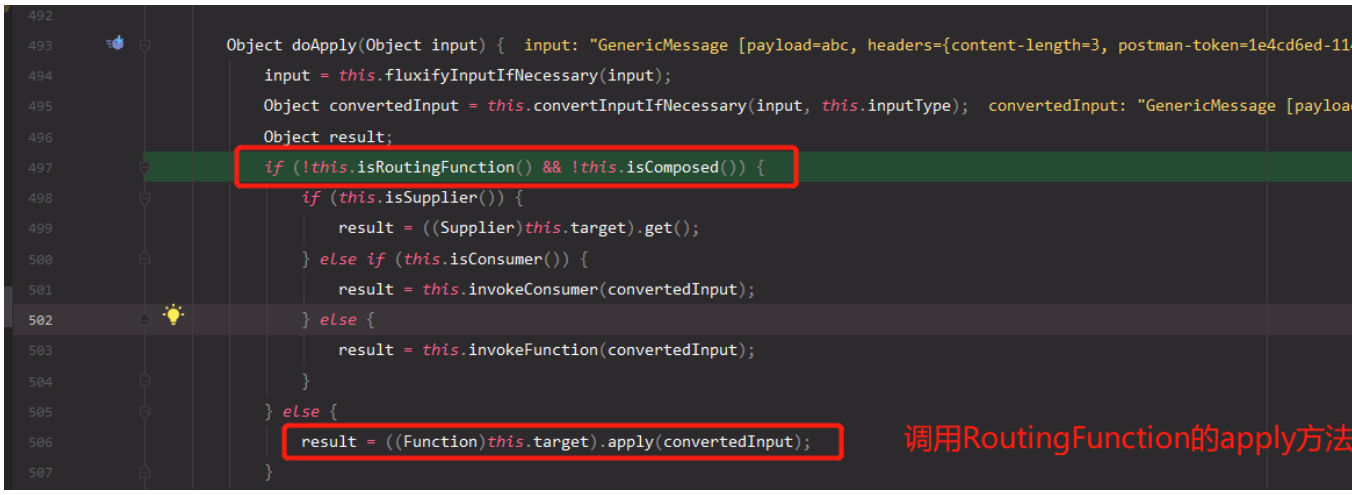
在 org.springframework.cloud.function.web.mvc.FunctionController#post 方法上下断点



程序会获取 body 中的参数，并传入 processRequest 方法中



程序会判断当前请求是否为 RoutingFunction，并将请求的内容和 Header 头编译成 Message 带入到 FunctionInvocationWrapper.apply 方法中，随后又进入其中的 doApply 方法



跟进 RoutingFunction 的 apply 方法

最后进入到

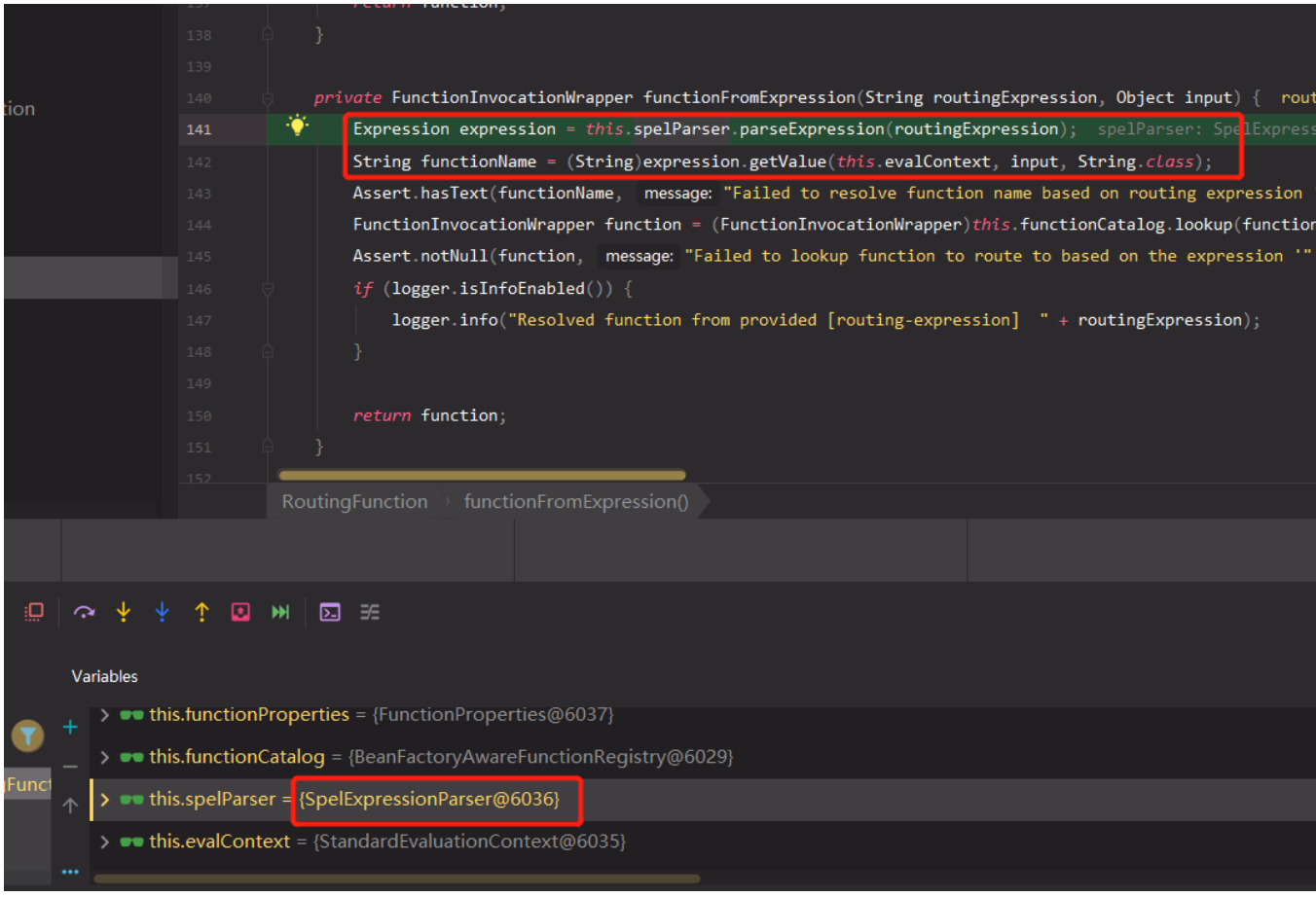
org.springframework.cloud.function.context.config.RoutingFunction#route 方法中

在这里判断了请求 headers 头中有没有 spring.cloud.function.routing-expression 参数

并将结果带入到 this.functionFromExpression()方法中

```
if (function == null) {  function: null
    if (StringUtils.hasText((String)message.getHeaders().get("spring.cloud.function.definition"))) {  message: "
        function = this.functionFromDefinition((String)message.getHeaders().get("spring.cloud.function.definitio
        if (function.isInputTypePublisher()) {
            this.assertOriginalInputIsNotPublisher(originalInputIsPublisher);
        }
    } else if (StringUtils.hasText((String)message.getHeaders().get("spring.cloud.function.routing-expression"))
        function = this.functionFromExpression((String)message.getHeaders().get("spring.cloud.function.routing-e
        if (function.isInputTypePublisher()) {
            this.assertOriginalInputIsNotPublisher(originalInputIsPublisher);
        }
    } else if (StringUtils.hasText(this.functionProperties.getRoutingExpression())) {
        function = this.functionFromExpression(this.functionProperties.getRoutingExpression(), message);
    } else {
        if (!StringUtils.hasText(this.functionProperties.getDefinition())) {
            throw new IllegalStateException("Failed to establish route, since neither were provided: 'spring.clo
```

最终直接由 SpelExpressionParser 来解析，导致 Spel 表达式注入。



从官方补丁对比可看出新增了 SimpleEvaluationContext，用于限制外部输入解析，

在解析前先判断 spring.cloud.function.routing-expression 的值是否取自 http 头，

外部输入使用 SimpleEvaluationContext，非外部输入使用 StandardEvaluationContext 进行解析

```
34    import org.springframework.expression.BeanResolver;
35    import org.springframework.expression.Expression;
36    import org.springframework.expression.spel.standard.SpelExpressionParser;
37  + import org.springframework.expression.spel.support.DataBindingPropertyAccessor;
38  + import org.springframework.expression.spel.support.SimpleEvaluationContext;
39    import org.springframework.expression.spel.support.StandardEvaluationContext;
40    import org.springframework.messaging.Message;
41    import org.springframework.util.Assert;
42    import org.springframework.util.StringUtils;
43
44    /**
45     * An implementation of Function which acts as a gateway/router by actually
46     * delegating incoming invocation to a function specified .. .
61
62        private final StandardEvaluationContext evalContext = new StandardEvaluationContext();
63
64  +     private final SimpleEvaluationContext headerEvalContext = SimpleEvaluationContext
65  +             .forPropertyAccessors(DataBindingPropertyAccessor.forReadOnlyAccess()).build();
66  +
67        private final SpelExpressionParser spelParser = new SpelExpressionParser();
68
69        private final FunctionCatalog functionCatalog;
70
71        private final FunctionProperties functionProperties;
72
```

应急修复是判断是否是从 Header 头中读取.否则进入安全函数

其中的 this.headerEvalContext 变量就是由 SimpleEvaluationContext 来解析的

```
133                         this.assertOriginalInputIsNotPublisher(originalInputIsPublisher);
134                     }

197         }
198
199         private FunctionInvocationWrapper functionFromExpression(String routingExpression, Object input) {
200 +           return functionFromExpression(routingExpression, input, false);
201 +       }
202 +
203 +       private FunctionInvocationWrapper functionFromExpression(String routingExpression, Object input, boolean isViaHead
204             Expression expression = spelParser.parseExpression(routingExpression);
205             if (input instanceof Message) {
206                 input = MessageUtils.toCaseInsensitiveHeadersStructure((Message<?>) input);
207             }
208
209 +           String functionName = isViaHeader ? expression.getValue(this.headerEvalContext, input, String.class) :
    expression.getValue(this.evalContext, input, String.class);
210             Assert.hasText(functionName, "Failed to resolve function name based on routing expression '" +
    functionProperties.getRoutingExpression() + "'");
211             FunctionInvocationWrapper function = functionCatalog.lookup(functionName);
212             Assert.notNull(function, "Failed to lookup function to route to based on the expression '"
```

修复建议

目前 Spring Cloud Function 官方已发布此漏洞的补丁，但尚未正式发布修复版本，建议相关用户尽快应用补丁进行防护并关注官方的更新版本。

补丁下载链接：

https://github.com/spring-cloud/spring-cloud-function/commit/0e89ee27b2e76138c16bcba6f4bca906c4f3744f

下载链接：

https://github.com/spring-cloud/spring-cloud-function/tags

在项目代码目录中执行以下命令获取 org.springframework.cloud:spring-cloud-function-context 组件版本，且组件版本在【3.0.0.RELEASE~3.2.2】之间

用户可排查应用程序中对 spring-cloud-function 组件的引用情况，并检查当前使用版本是否受到影响。如果程序使用 Maven 打包，可查看项目的 pom.xml 文件中是否引入相关组件。