**聊聊spring cloud gateway的ForwardedHeadersFilter**

[go4it](https://www.jianshu.com/u/288675f429fa)关注

2018.06.02 14:25:52字数 317阅读 1,284

**序**

本文主要研究一下spring cloud gateway的ForwardedHeadersFilter

**GatewayAutoConfiguration**

spring-cloud-gateway-core-2.0.0.RC1-sources.jar!/org/springframework/cloud/gateway/config/GatewayAutoConfiguration.java

@Configuration

@ConditionalOnProperty(name = "spring.cloud.gateway.enabled", matchIfMissing = true)

@EnableConfigurationProperties

@AutoConfigureBefore(HttpHandlerAutoConfiguration.class)

@AutoConfigureAfter({GatewayLoadBalancerClientAutoConfiguration.class, GatewayClassPathWarningAutoConfiguration.class})

@ConditionalOnClass(DispatcherHandler.class)

public class GatewayAutoConfiguration {

//......

@Bean

@ConditionalOnProperty(name = "spring.cloud.gateway.forwarded.enabled", matchIfMissing = true)

public ForwardedHeadersFilter forwardedHeadersFilter() {

return new ForwardedHeadersFilter();

}

//......

}

**ForwardedHeadersFilter**

spring-cloud-gateway-core-2.0.0.RC1-sources.jar!/org/springframework/cloud/gateway/filter/headers/ForwardedHeadersFilter.java

public class ForwardedHeadersFilter implements HttpHeadersFilter, Ordered {

public static final String FORWARDED\_HEADER = "Forwarded";

@Override

public int getOrder() {

return 0;

}

@Override

public HttpHeaders filter(HttpHeaders input, ServerWebExchange exchange) {

ServerHttpRequest request = exchange.getRequest();

HttpHeaders original = input;

HttpHeaders updated = new HttpHeaders();

// copy all headers except Forwarded

original.entrySet().stream()

.filter(entry -> !entry.getKey().toLowerCase().equalsIgnoreCase(FORWARDED\_HEADER))

.forEach(entry -> updated.addAll(entry.getKey(), entry.getValue()));

List<Forwarded> forwardeds = parse(original.get(FORWARDED\_HEADER));

for (Forwarded f : forwardeds) {

updated.add(FORWARDED\_HEADER, f.toString());

}

//TODO: add new forwarded

URI uri = request.getURI();

String host = original.getFirst(HttpHeaders.HOST);

Forwarded forwarded = new Forwarded()

.put("host", host)

.put("proto", uri.getScheme());

InetSocketAddress remoteAddress = request.getRemoteAddress();

if (remoteAddress != null) {

String forValue = remoteAddress.getAddress().getHostAddress();

int port = remoteAddress.getPort();

if (port >= 0) {

forValue = forValue + ":" + port;

}

forwarded.put("for", forValue);

}

// TODO: support by?

updated.add(FORWARDED\_HEADER, forwarded.toHeaderValue());

return updated;

}

/\* for testing \*/ static List<Forwarded> parse(List<String> values) {

ArrayList<Forwarded> forwardeds = new ArrayList<>();

if (CollectionUtils.isEmpty(values)) {

return forwardeds;

}

for (String value : values) {

Forwarded forwarded = parse(value);

forwardeds.add(forwarded);

}

return forwardeds;

}

/\* for testing \*/ static Forwarded parse(String value) {

String[] pairs = StringUtils.tokenizeToStringArray(value, ";");

LinkedCaseInsensitiveMap<String> result = splitIntoCaseInsensitiveMap(pairs);

if (result == null) return null;

Forwarded forwarded = new Forwarded(result);

return forwarded;

}

@Nullable

/\* for testing \*/ static LinkedCaseInsensitiveMap<String> splitIntoCaseInsensitiveMap(String[] pairs) {

if (ObjectUtils.isEmpty(pairs)) {

return null;

}

LinkedCaseInsensitiveMap<String> result = new LinkedCaseInsensitiveMap<>();

for (String element : pairs) {

String[] splittedElement = StringUtils.split(element, "=");

if (splittedElement == null) {

continue;

}

result.put(splittedElement[0].trim(), splittedElement[1].trim());

}

return result;

}

}

这个filter首先拷贝了请求的header，然后将请求中的Forwarded提取出来，解析成一个个Forwarded对象，添加到新的HttpHeaders中。除此之外，还补充了一个转发信息的Forwarded(host,proto,for)

**Forwarded**

**语法**

Forwarded: by=<identifier>; for=<identifier>; host=<host>; proto=<http|https>

* by=<identifier> 该请求进入到代理服务器的接口。
* for=<identifier> 发起请求的客户端以及代理链中的一系列的代理服务器。
* host=<host> 代理接收到的 Host首部的信息。
* proto=<http|https> 表示发起请求时采用的何种协议（通常是 "http" 或者 "https"）。

**实例**

Forwarded: for=192.0.2.60; proto=http; by=203.0.113.43

Forwarded: proto=http;host="localhost:10000";for="0:0:0:0:0:0:0:1:56443"

**对象**

static class Forwarded {

private static final char EQUALS = '=';

private static final char SEMICOLON = ';';

private final Map<String, String> values;

public Forwarded() {

this.values = new HashMap<>();

}

public Forwarded(Map<String, String> values) {

this.values = values;

}

public Forwarded put(String key, String value) {

this.values.put(key, quoteIfNeeded(value));

return this;

}

private String quoteIfNeeded(String s) {

if (s.contains(":")) { //TODO: broaded quote

return "\""+s+"\"";

}

return s;

}

public String get(String key) {

return this.values.get(key);

}

/\* for testing \*/ Map<String, String> getValues() {

return this.values;

}

@Override

public String toString() {

return "Forwarded{" +

"values=" + this.values +

'}';

}

public String toHeaderValue() {

StringBuilder builder = new StringBuilder();

for (Map.Entry<String, String> entry : this.values.entrySet()) {

if (builder.length() > 0) {

builder.append(SEMICOLON);

}

builder.append(entry.getKey())

.append(EQUALS)

.append(entry.getValue());

}

return builder.toString();

}

}

**小结**

RFC 7239(June 2014)提出了一个标准化的Forwarded头部，来携带反向代理的基本信息，用于替代X-Forwarded系列及X-Real-IP等非标准化的头部。而ForwardedHeadersFilter便是提供了Forwarded头部的转发支持，目前经过gateway的请求会带上一个转发信息的Forwarded(host,proto,for)。

**doc**

* [Forwarded](https://developer.mozilla.org/zh-CN/docs/Web/HTTP/Headers/Forwarded)
* [Forwarded HTTP Extension](https://tools.ietf.org/html/rfc7239)
* [THE FORWARDED HEADER](https://www.shimmercat.com/en/docs/1.5/forwarded-header/)
* [Using the Forwarded header](https://www.nginx.com/resources/wiki/start/topics/examples/forwarded/)