web 服务器请求走私漏洞

谢远峰

天津大学-智能与计算学部-网络安全

2024-09-09

1. 背景与问题

1. 背景与问题

目标

- web 服务器请求走私漏洞：HTTP 请求的歧义性解析

典型例子：HTTP/1.1 协议中定义了两种指示请求体长度的方法:Content-Length 头和 Transfer-Encoding 头。当这两个头同时出现时,Transfer-Encoding 应该优先。然而,不同的服务器可能会有不同的实现。

攻击目标：CL.TE 类型走私。前端服务器可能优先考虑 Content-Length；后端服务器可能优先考虑 Transfer-Encoding。

攻击方法：Content-Length 值小于实际请求体长度；Transfer-Encoding: chunked 后跟着一个畸形的 chunk

目标

- web 服务器请求走私漏洞：HTTP 请求的歧义性解析

典型例子：HTTP/1.1 协议中定义了两种指示请求体长度的方法:Content-Length 头和 Transfer-Encoding 头。当这两个头同时出现时,Transfer-Encoding 应该优先。然而,不同的服务器可能会有不同的实现。

攻击目标：CL.TE 类型走私。前端服务器可能优先考虑 Content-Length；后端服务器可能优先考虑 Transfer-Encoding。

攻击方法：Content-Length 值小于实际请求体长度；Transfer-Encoding: chunked 后跟着一个畸形的 chunk

- RFC 相关文档资料参考

Field definitions（RFC2822-3.6）：http 消息头部的结构和内容

Obsolete header fields（RFC2822-4.5）：保持向后兼容性,允许处理旧格式 message

HTTP Message-Message Headers（RFC2616-4.2）：http message header 定义

Protocol Specific Meta Variables（RFC3875-4.1.18）: CGI 变量的定义

RFC 文档冲突

- 多个相同名称的头字段处理：
  - RFC2616：对于相同名称的多个头字段，代理不应改变这个顺序。这些字段可以组合成一个以逗号分隔的列表。
  - RFC 3875：服务器必须将多个相同名称的头字段重写为相同语义的单个值。
  - 冲突: RFC 2616 保留头字段的顺序和独立性，而 RFC 3875 要求将它们合并为一个单一值。
-
-
-
-

RFC 文档冲突

- 多个相同名称的头字段处理：
  - RFC2616：对于相同名称的多个头字段，代理不应改变这个顺序。这些字段可以组合成一个以逗号分隔的列表。
  - RFC 3875：服务器必须将多个相同名称的头字段重写为相同语义的单个值。
  - 冲突: RFC 2616 保留头字段的顺序和独立性，而 RFC 3875 要求将它们合并为一个单一值。
- 跨多行的头字段处理：
  - RFC 2616 允许头字段跨多行，只要后续行以空格或制表符开头。
  - RFC 3875 明确要求将跨多行的头字段合并到一行。
-
-
-

RFC 文档冲突

- 多个相同名称的头字段处理：
  - RFC2616：对于相同名称的多个头字段，代理不应改变这个顺序。这些字段可以组合成一个以逗号分隔的列表。
  - RFC 3875：服务器必须将多个相同名称的头字段重写为相同语义的单个值。
  - 冲突: RFC 2616 保留头字段的顺序和独立性，而 RFC 3875 要求将它们合并为一个单一值。
- 跨多行的头字段处理：
  - RFC 2616 允许头字段跨多行，只要后续行以空格或制表符开头。
  - RFC 3875 明确要求将跨多行的头字段合并到一行。
- 大小写敏感性：
  - RFC 2616 规定字段名称不区分大小写。
  - RFC 3875 在转换为 CGI 环境变量时，要求将头字段名称转换为大写。
-
-

RFC 文档冲突

- 多个相同名称的头字段处理：
  - RFC2616：对于相同名称的多个头字段，代理不应改变这个顺序。这些字段可以组合成一个以逗号分隔的列表。
  - RFC 3875：服务器必须将多个相同名称的头字段重写为相同语义的单个值。
  - 冲突: RFC 2616 保留头字段的顺序和独立性，而 RFC 3875 要求将它们合并为一个单一值。
- 跨多行的头字段处理：
  - RFC 2616 允许头字段跨多行，只要后续行以空格或制表符开头。
  - RFC 3875 明确要求将跨多行的头字段合并到一行。
- 大小写敏感性：
  - RFC 2616 规定字段名称不区分大小写。
  - RFC 3875 在转换为 CGI 环境变量时，要求将头字段名称转换为大写。
- 头字段的选择性移除：
  - RFC 2616 没有特别提到移除特定的头字段。
  - RFC 3875 建议移除某些头字段，如授权信息和仅与客户端通信相关的字段。
-

RFC 文档冲突

- 多个相同名称的头字段处理：
  - RFC2616：对于相同名称的多个头字段，代理不应改变这个顺序。这些字段可以组合成一个以逗号分隔的列表。
  - RFC 3875：服务器必须将多个相同名称的头字段重写为相同语义的单个值。
  - 冲突: RFC 2616 保留头字段的顺序和独立性，而 RFC 3875 要求将它们合并为一个单一值。
- 跨多行的头字段处理：
  - RFC 2616 允许头字段跨多行，只要后续行以空格或制表符开头。
  - RFC 3875 明确要求将跨多行的头字段合并到一行。
- 大小写敏感性：
  - RFC 2616 规定字段名称不区分大小写。
  - RFC 3875 在转换为 CGI 环境变量时，要求将头字段名称转换为大写。
- 头字段的选择性移除：
  - RFC 2616 没有特别提到移除特定的头字段。
  - RFC 3875 建议移除某些头字段，如授权信息和仅与客户端通信相关的字段。
- 语义保留 vs 表示更改：
  - RFC 2616 强调在转发消息时不应更改语义。
  - RFC 3875 允许服务器在必要时更改数据的表示（例如字符集），以适应 CGI 元变量。

Web 服务器网关接口-wsgrief

- wsgiref 是 WSGI 规范的参考实现，可以用于将 WSGI 支持添加到 Web 服务器或框架中。wsgiref 提供了一套完整的工具，用于开发、测试和验证 WSGI 应用程序和服务器。
-
-

Web 服务器网关接口-wsgrief

- wsgiref 是 WSGI 规范的参考实现，可以用于将 WSGI 支持添加到 Web 服务器或框架中。wsgiref 提供了一套完整的工具，用于开发、测试和验证 WSGI 应用程序和服务器。
- 提供了操作 WSGI 环境变量和响应头的实用程序、用于实现 WSGI 服务器的基类、一个演示 HTTP 服务器以及静态类型检查的类型和用于检查 WSGI 服务器和应用程序是否符合 WSGI 规范的验证工具。
- 实例
  - wsgiref.util(WSGI 环境实用工具)
  - wsgiref.headers(WSGI 响应头工具)
  - wsgiref.simple_server(WSGI HTTP 服务器样例)
  - wsgiref.validate(WSGI 规范一致性检查)
  - wsgiref.handlers(服务器/网关基类)
  - wsgiref.types(用于静态类型检查的 WSGI 类型)

漏洞实例-Gunicorn 的 WSGI 头部欺骗漏洞

Gunicorn 在处理 HTTP 头部时，会将头部名中的破折号（dash）转换为下划线（underscore）。例如，Foo-Bar 和 Foo_Bar 都会被转换为 HTTP_FOO_BAR。此外，如果 HTTP 请求同时包含 Foo-Bar: a 和 Foo_Bar: b，Gunicorn 会将它们合并为 {‘HTTP_FOO_BAR’: ‘a,b’}。

漏洞实例-Gunicorn 的 WSGI 头部欺骗漏洞

Gunicorn 在处理 HTTP 头部时，会将头部名中的破折号（dash）转换为下划线（underscore）。例如，Foo-Bar 和 Foo_Bar 都会被转换为 HTTP_FOO_BAR。此外，如果 HTTP 请求同时包含 Foo-Bar: a 和 Foo_Bar: b，Gunicorn 会将它们合并为 {'HTTP_FOO_BAR': 'a,b'}。

安全漏洞隐患：

安全敏感头部的绕过： 如果前端代理会删除任何传入的 X-Auth-User 头部值，那么攻击者可以传入一个 X-Auth_User 头部（带下划线），绕过这种保护机制。

漏洞实例-Gunicorn 的 WSGI 头部欺骗漏洞

Gunicorn 在处理 HTTP 头部时，会将头部名中的破折号（dash）转换为下划线（underscore）。例如，Foo-Bar 和 Foo_Bar 都会被转换为 HTTP_FOO_BAR。此外，如果 HTTP 请求同时包含 Foo-Bar: a 和 Foo_Bar: b，Gunicorn 会将它们合并为 {'HTTP_FOO_BAR': 'a,b'}。

安全漏洞隐患：

安全敏感头部的绕过： 如果前端代理会删除任何传入的 X-Auth-User 头部值，那么攻击者可以传入一个 X-Auth_User 头部（带下划线），绕过这种保护机制。

IP 伪造：如果攻击者设置 X_Forwarded_For: 5.6.7.8（伪造的 IP），而前端代理设置 X-Forwarded-For: 1.2.3.4（真实 IP），Python 应用程序可能会看到 1.2.3.4,5.6.7.8 或 5.6.7.8,1.2.3.4，具体取决于代理的头部顺序。

漏洞实例-Gunicorn 的 WSGI 头部欺骗漏洞

Gunicorn 在处理 HTTP 头部时，会将头部名中的破折号（dash）转换为下划线（underscore）。例如，Foo-Bar 和 Foo_Bar 都会被转换为 HTTP_FOO_BAR。此外，如果 HTTP 请求同时包含 Foo-Bar: a 和 Foo_Bar: b，Gunicorn 会将它们合并为 {'HTTP_FOO_BAR': 'a,b'}。

安全漏洞隐患：

安全敏感头部的绕过：如果前端代理会删除任何传入的 X-Auth-User 头部值，那么攻击者可以传入一个 X-Auth_User 头部（带下划线），绕过这种保护机制。

IP 伪造：如果攻击者设置 X_Forwarded_For: 5.6.7.8（伪造的 IP），而前端代理设置 X-Forwarded-For: 1.2.3.4（真实 IP），Python 应用程序可能会看到 1.2.3.4,5.6.7.8 或 5.6.7.8,1.2.3.4，具体取决于代理的头部顺序。

修复方法：

- 始终忽略包含下划线的头部：这是一个不好的主意，因为许多 Gunicorn 用户依赖当前的行为。
- 添加类似 Nginx 的 underscores_in_headers 选项，默认忽略这些头部：这是一个较好的选择，但可能会导致重大兼容性问题，并引起混淆。
- 添加类似 Nginx 的 underscores_in_headers 选项，默认允许这些头部：尽管有些用户可以选择加入，但这依然是一个不够完美的解决方案。