(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 112287349 A
(43)申请公布日 2021.01.29

(21)申请号 201910681557.9

(22)申请日 2019.07.25

(71)申请人 腾讯科技(深圳)有限公司
地址 518057 广东省深圳市南山区高新区科技中一路腾讯大厦35层

(72)发明人 吴斐 蒋剑琴 黄梓群 李斌

(74)专利代理机构 广州三环专利商标代理有限公司 44202
代理人 郝传鑫 熊永强

(51)Int.Cl.
G06F 21/57(2013.01)

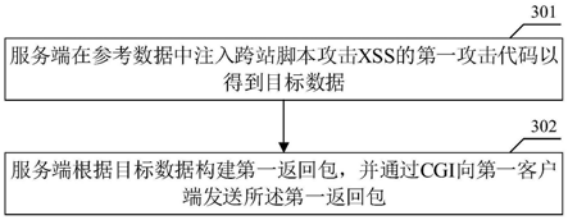
权利要求书2页 说明书17页 附图11页

(54)发明名称

安全漏洞检测方法及服务端

(57)摘要

本发明实施例公开了一种安全漏洞检测方法、服务端及存储介质，该方法可包括：服务端在参考数据中注入跨站脚本攻击XSS的第一攻击代码以得到目标数据；所述参考数据包括所述服务端待通过通用网关接口CGI向第一客户端发送的数据；所述服务端根据所述目标数据构建第一返回包，并通过所述CGI向所述第一客户端发送所述第一返回包，所述第一返回包用于检测XSS漏洞。本发明中，服务端将待通过CGI向第一客户端发送的数据中注入第一攻击代码以覆盖第一客户端所有引用CGI层的变量，能够提高测试XSS漏洞的过程中XSS的攻击代码注入的覆盖率，进而测试各种类型的XSS漏洞。



CN 112287349 A

1.一种安全漏洞检测方法,其特征在于,包括:

服务端在参考数据中注入跨站脚本攻击XSS的第一攻击代码以得到目标数据;所述参考数据包括所述服务端待通过通用网关接口CGI向第一客户端发送的数据;

所述服务端根据所述目标数据构建第一返回包,并通过所述CGI向所述第一客户端发送所述第一返回包,所述第一返回包用于检测XSS漏洞。

2.根据权利要求1所述的方法,其特征在于,所述服务端在参考数据中注入跨站脚本攻击XSS的第一攻击代码以得到目标数据之前,所述方法还包括:

所述服务端接收来自所述第一客户端的第一访问请求,所述第一访问请求用于获取所述参考数据;

所述服务端从存储器或数据库获取针对所述第一访问请求的所述参考数据。

3.根据权利要求2所述的方法,其特征在于,所述第一访问请求包括第一标识,所述第一标识用于指示所述第一客户端为用于检测XSS漏洞的客户端。

4.根据权利要求2所述的方法,其特征在于,所述服务端接收来自所述第一客户端的第一访问请求包括:

所述服务端接收所述第一客户端通过浏览器发送的所述第一访问请求;所述第一返回包用于在所述浏览器显示目标页面,所述目标页面用于检测XSS漏洞。

5.根据权利要求2所述的方法,其特征在于,所述服务端接收来自所述第一客户端的第一访问请求包括:

所述服务端接收所述第一客户端通过测试工具发送的所述第一访问请求;所述测试工具运行于所述第一客户端,且用于根据所述第一返回包检测XSS漏洞。

6.根据权利要求2至5任一项所述的方法,其特征在于,所述服务端在参考数据中注入跨站脚本攻击XSS的第一攻击代码以得到目标数据包括:

所述服务端将所述参考数据中的至少一个字符串类型的数据替换为所述第一攻击代码以得到所述目标数据。

7.根据权利要求6所述的方法,其特征在于,所述方法还包括:

所述服务端在初始数据中注入XSS的第二攻击代码以得到中间数据;所述初始数据包括所述服务端待通过所述CGI向第二客户端发送的数据;所述第二攻击代码与所述第一攻击代码不同;

所述服务端根据所述中间数据构建第二返回包,并通过所述CGI向所述第二客户端发送所述第二返回包,所述第二返回包用于检测XSS漏洞。

8.根据权利要求7所述的方法,其特征在于,所述第一标识用于指示所述第一客户端为用于检测第一XSS漏洞的客户端,所述第一标识对应所述第一攻击代码;所述服务端在初始数据中注入XSS的第二攻击代码以得到中间数据之前,所述方法还包括:

所述服务端接收来自所述第二客户端的第二访问请求,所述第二访问请求用于获取所述初始数据;所述第二访问请求包括第二标识,所述第二标识用于指示所述第二客户端为用于检测第二XSS漏洞的客户端;所述第二标识对应所述第二攻击代码;

所述服务端从所述存储器或所述数据库获取针对所述第二访问请求的所述初始数据。

9.一种服务端,其特征在于,包括:

代码注入模块,用于在参考数据中注入跨站脚本攻击XSS的第一攻击代码以得到目标

数据;所述参考数据包括所述服务端待通过通用网关接口CGI向第一客户端发送的数据;

所述CGI,用于根据所述目标数据构建第一返回包,并向所述第一客户端发送所述第一返回包,所述第一返回包用于检测XSS漏洞。

10.根据权利要求9所述的服务端,其特征在于,所述服务端还包括:

接收模块,用于接收来自所述第一客户端的第一访问请求,所述第一访问请求用于获取所述参考数据;

所述CGI,还用于从存储器或数据库获取针对所述第一访问请求的所述参考数据。

安全漏洞检测方法及服务端

技术领域

[0001] 本发明涉及计算机安全领域，具体涉及XSS漏洞检测技术领域，尤其涉及一种安全漏洞检测方法及服务端。

背景技术

[0002] 跨站脚本攻击(Cross Site Scripting，安全专家们通常将其缩写成XSS)指的是恶意攻击者往万维网(Web)页面里插入恶意超文本标记语言(HyperText Markup Language，HTLM)代码，当用户浏览该页面之时，嵌入其中Web页面里面的HTLM代码会被执行，从而达到恶意用户的特殊目的。在实际应用中，用户在浏览网站、使用即时通讯软件、甚至在阅读电子邮件时，通常会点击其中的链接。攻击者通过在链接中插入恶意代码，就能够盗取用户信息。攻击者通常会用十六进制(或其他编码方式)将链接编码，以免用户怀疑它的合法性。网站在接收到包含恶意代码的请求之后会产成一个包含恶意代码的页面，而这个页面看起来就像是那个网站应当生成的合法页面一样。许多流行的留言本和论坛程序允许用户发表包含HTML的帖子。假设用户甲发表了一篇包含恶意脚本的帖子，那么用户乙在浏览这篇帖子时，恶意脚本就会执行，盗取用户乙的会话(session)信息。

[0003] XSS漏洞是Web应用程序中最常见的漏洞之一。如果一个站点(例如Web服务器运行的网站)没有预防XSS漏洞的方法，那么就存在XSS漏洞，难以防御XSS。目前，检测XSS漏洞的常用方式是替换通用网关接口(Common Gateway Interface，CGI)的输入参数值，即在链接中插入恶意代码，检测该链接返回的页面是否有恶意代码。CGI在物理上是一段程序，运行在服务器上，提供同客户端HTML页面交互的接口。然而，这种方式仅能检测从CGI注入恶意代码的XSS漏洞，而不能准确地检测到其他类型的XSS漏洞，例如存储型XSS漏洞。

发明内容

[0004] 本发明实施例提供了一种安全漏洞检测方法及服务端，可涵盖CGI流程中所有的外部引入变量，能够提高在测试XSS漏洞的过程中XSS注入的覆盖率。

[0005] 第一方面，本发明实施例提供了一种安全漏洞检测方法，该方法可包括：服务端在参考数据中注入跨站脚本攻击XSS的第一攻击代码以得到目标数据；所述参考数据包括所述服务端待通过通用网关接口CGI向第一客户端发送的数据；所述服务端根据所述目标数据构建第一返回包，并通过所述CGI向所述第一客户端发送所述第一返回包，所述第一返回包用于检测XSS漏洞。

[0006] 本发明实施例中，服务端将待通过CGI向第一客户端发送的数据中注入第一攻击代码以覆盖第一客户端所有引用CGI层的变量，能够提高测试XSS漏洞的过程中XSS的攻击代码注入的覆盖率，进而测试各种类型的XSS漏洞。

[0007] 在一个可选的实现方式中，所述服务端在参考数据中注入跨站脚本攻击XSS的第一攻击代码以得到目标数据之前，所述方法还包括：

[0008] 所述服务端接收来自所述第一客户端的第一访问请求，所述第一访问请求用于获

取所述参考数据；

[0009] 所述服务端从存储器或数据库获取针对所述第一访问请求的所述参考数据。

[0010] 在该实现方式中，服务端根据来自第一客户端的第一访问请求，从存储器或数据库获取相应的数据，操作简单。

[0011] 在一个可选的实现方式中，所述服务端接收来自所述第一客户端的第一访问请求包括：

[0012] 所述CGI接收来自所述第一客户端的所述第一访问请求；

[0013] 所述服务端从存储器或数据库获取针对所述第一访问请求的所述参考数据包括：

[0014] 所述CGI向代码注入模块发送数据获取请求；所述数据获取请求用于请求所述参考数据；

[0015] 所述代码注入模块从所述存储器或所述数据库获取所述参考数据；

[0016] 所述服务端在参考数据中注入跨站脚本攻击XSS的第一攻击代码以得到目标数据包括：

[0017] 所述代码注入模块在所述参考数据中注入第一攻击代码以得到所述目标数据。

[0018] 由于某种原因(比如对象创建的开销很大，或者某些操作需要安全控制，或者需要进程外的访问等)，CGI直接访问存储器或者数据库(即存储后台)会带来很多麻烦，这时就可以在CGI和存储后台之间增加一个中间代理层。代码注入模块既可以在参考数据中注入攻击代码，又能起到中间代理层(例如proxy层)的作用。

[0019] 在该实现方式中，CGI通过代码注入模块从存储器或数据库获取参考数据，相当于在CGI和存储后台之间多了一层中间代理层，可以提高数据获取速度，并且能够避免CGI直接访问存储后台造成的其他问题。

[0020] 在一个可选的实现方式中，所述CGI向代码注入模块发送数据获取请求之前，所述方法还包括：

[0021] 所述CGI向所述代码注入模块发送攻击代码获取请求，所述攻击代码获取请求用于请求所述第一攻击代码；

[0022] 所述CGI接收来自所述代码注入模块的所述第一攻击代码；

[0023] 所述CGI向代码注入模块发送数据获取请求包括：

[0024] 所述CGI向所述代码注入模块发送携带有所述第一攻击代码或者所述第一攻击代码的标识的所述数据获取请求。

[0025] 在该实现方式中，CGI向代码注入模块发送携带有第一攻击代码或者该第一攻击代码的标识的数据获取请求，该代码注入模块将注入该第一攻击代码的参数数据返回给CGI，代码注入模块可以准确地注入CGI所指示的攻击代码。在该实现方式中，CGI在获取到该第一攻击代码后，可以将该第一攻击代码或者该第一攻击代码的标识携带在向代码注入模块发送的多个数据获取请求中，而不必每次通过该代码注入模块获取数据时均获取一次攻击代码，可以减少不必要的操作。

[0026] 在一个可选的实现方式中，所述代码注入模块在所述参考数据中注入第一攻击代码以得到所述目标数据之前，所述方法还包括：

[0027] 所述代码注入模块从所述存储器或所述数据库获取参考定义，所述参考定义用于指示所述参考数据中各字符的类型；

[0028] 所述代码注入模块在所述参考数据中注入第一攻击代码以得到所述目标数据包括：

[0029] 根据所述参考定义，将所述参考数据中字符串类型的数据替换为所述第一攻击代码以得到所述目标数据。

[0030] 在该实现方式中，根据参考定义可以准确、快速地确定该参考数据中各字符串类型的数据，以便于准确地注入攻击代码。

[0031] 在一个可选的实现方式中，所述第一访问请求包括第一标识，所述第一标识用于指示所述第一客户端为用于检测XSS漏洞的客户端。

[0032] 服务端执行的一项XSS漏洞测试任务是接收来自第一客户端的第一访问请求；从存储器或数据库获取针对该第一访问请求的所参考数据；根据目标数据构建第一返回包，并通过CGI向所述第一客户端发送该第一返回包。该服务端还可以执行除该项XSS漏洞测试任务之外的任务。在实际应用中，服务端可以与一部分客户端(第一访问请求中包括第一标识的第一客户端)进行交互来测试XSS漏洞，也可以处理其他客户端的访问请求(不包括该第一标识)来执行其他任务。也就是说，服务端不限于XSS漏洞测试任务，还可以执行其他任务，例如其他测试任务。

[0033] 在一个可选的实现方式中，所述服务端接收来自所述第一客户端的第一访问请求包括：所述服务端接收所述第一客户端通过浏览器发送的所述第一访问请求；所述第一返回包用于在所述浏览器显示目标页面，所述目标页面用于检测XSS漏洞。

[0034] 服务端可以通过CGI接收第一客户端通过浏览器发送的第一访问请求。该目标页面可包括浏览器执行XSS的攻击代码得到的内容，例如弹框。在目标页面包括恶意内容的情况下，可以认为存在XSS漏洞；在该目标页面未包括恶意内容的情况下，可以认为不存在XSS漏洞。恶意内容是指浏览器执行XSS的攻击代码得到的内容。可以理解，若目标页面不包括恶意内容，则表示不存在XSS漏洞，即浏览器和服务端可以有效预防XSS；否则，表示存在XSS漏洞，即浏览器和服务端不能有效预防XSS，需要加强预防XSS的措施。

[0035] 在该实现方式中，服务端向第一客户端发送用于在该第一客户端的浏览器显示目标页面的第一返回包，以便于测试是否存储XSS漏洞，实现简单。

[0036] 在一个可选的实现方式中，所述服务端接收来自所述第一客户端的第一访问请求包括：所述服务端接收所述第一客户端通过测试工具发送的所述第一访问请求；所述测试工具运行于所述第一客户端，且用于根据所述第一返回包检测XSS漏洞。

[0037] 该测试工具可以是运行于第一客户端的专门用于检测XSS漏洞的自动化测试工具。第一客户端通过测试工具发送第一访问请求可以是该测试工具模拟用户点击页面或链接、页面滚动、键盘输入等常规用户操作(对应于发送第一访问请求)，以便于触发页面交互逻辑，即与服务端进行交互的操作。测试工具可以对第一返回包(即页面源码)进行扫描以查找第一返回包中必须要处理的特定代码段(即XSS的攻击代码)；如果在扫描中发现该第一返回包中残留这样的代码，则认为存在XSS漏洞，否则，认为不存在XSS漏洞。该测试工具在检测到特定代码段后，可以记录本次的检测信息，例如检测的时间、点击的链接等；也可以生成一份测试报告，该测试报告用于说明检测到的XSS的情况。可以理解，测试工具可以自动向服务端发送第一访问请求以触发页面交互逻辑，并对交互后的页面源码进行扫描以检测XSS漏洞，测试效率高、操作简单。

[0038] 在该实现方式中,服务端向第一客户端发送第一返回包,以便于运行于该第一客户端上的测试工具根据该第一返回包来检测XSS漏洞,可以有效提高测试效率。

[0039] 在一个可选的实现方式中,所述服务端在参考数据中注入跨站脚本攻击XSS的第一攻击代码以得到目标数据包括:所述服务端将所述参考数据中的至少一个字符串类型的数据替换为所述第一攻击代码以得到所述目标数据。

[0040] XSS均是将第一客户端请求获取的数据中的字符串类型的数据替换为第一攻击代码。在该实现方式中,通过将参考数据中的字符串类型的替换为攻击代码,可以模拟任何类型的XSS,实现简单。

[0041] 在一个可选的实现方式中,所述方法还包括:所述服务端在初始数据中注入XSS的第二攻击代码以得到中间数据;所述初始数据包括所述服务端待通过所述CGI向第二客户端发送的数据;所述第二攻击代码与所述第一攻击代码不同;所述服务端根据所述中间数据构建第二返回包,并通过所述CGI向所述第二客户端发送所述第二返回包,所述第二返回包用于检测XSS漏洞。

[0042] 在该实现方式中,服务端可以向不同客户端请求的数据注入不同的攻击代码,以便于同时检测不同类型的XSS漏洞,提高检测效率。

[0043] 在一个可选的实现方式中,所述第一标识用于指示所述第一客户端为用于检测第一XSS漏洞的客户端,所述第一标识对应所述第一攻击代码;所述服务端在初始数据中注入XSS的第二攻击代码以得到中间数据之前,所述方法还包括:所述服务端接收来自所述第二客户端的第二访问请求,所述第二访问请求用于获取所述初始数据;所述第二访问请求包括第二标识,所述第二标识用于指示所述第二客户端为用于检测第二XSS漏洞的客户端;所述第二标识对应所述第二攻击代码;所述服务端从所述存储器或所述数据库获取针对所述第二访问请求的所述初始数据。

[0044] 在该实现方式中,服务端根据各客户端的标识,对各客户端请求的数据注入相应的XSS攻击代码,可以同时检测不同类型的XSS漏洞,提高检测效率、实现简单。

[0045] 在一个可选的实现方式中,所述服务端根据所述目标数据构建第一返回包之前,所述方法还包括:所述服务端过滤所述目标数据中会导致所述第一攻击代码对应的脚本执行的内容,和/或,对所述目标数据进行HTML编码。

[0046] 服务端运行有至少一种过滤会导致XSS脚本执行的相关内容的方法,和/或,至少一种对动态输出到页面的内容进行HTML编码的方式。在该方式中,在根据目标数据构建返回包之前,执行至少一种XSS防御策略,进而验证在这些XSS防御策略是否能有效防御XSS。

[0047] 第二方面,本发明实施例提供了一种服务端,该服务端可包括:代码注入模块,用于在参考数据中注入跨站脚本攻击XSS的第一攻击代码以得到目标数据;所述参考数据包括所述服务端待通过通用网关接口CGI向第一客户端发送的数据;所述CGI,用于根据所述目标数据构建第一返回包,并向所述第一客户端发送所述第一返回包,所述第一返回包用于检测XSS漏洞。该服务端可以是Web服务器或者其他类型的服务器。

[0048] 在一个可选的实现方式中,所述服务端还包括:接收模块,用于接收来自所述第一客户端的第一访问请求,所述第一访问请求用于获取所述参考数据;所述CGI,还用于从存储器或数据库获取针对所述第一访问请求的所述参考数据。所述接口模块可以是所述CGI。

[0049] 在一个可选的实现方式中,所述CGI,具体用于CGI接收来自所述第一客户端的所

述第一访问请求;向代码注入模块发送数据获取请求;所述数据获取请求用于请求所述参考数据;

[0050] 所述代码注入模块,具体用于从所述存储器或所述数据库获取所述参考数据;在所述参考数据中注入第一攻击代码以得到所述目标数据。

[0051] 在一个可选的实现方式中,所述CGI,还用于向所述代码注入模块发送攻击代码获取请求,所述攻击代码获取请求用于请求所述第一攻击代码;接收来自所述代码注入模块的所述第一攻击代码;向所述代码注入模块发送携带有所述第一攻击代码或者所述第一攻击代码的标识的所述数据获取请求。

[0052] 在一个可选的实现方式中,所述代码注入模块,还用于从所述存储器或所述数据库获取参考定义,所述参考定义用于指示所述参考数据中各字符的类型;根据所述参考定义,将所述参考数据中字符串类型的数据替换为所述第一攻击代码以得到所述目标数据。

[0053] 在一个可选的实现方式中,所述第一访问请求包括第一标识,所述第一标识用于指示所述第一客户端为用于检测XSS漏洞的客户端。

[0054] 在一个可选的实现方式中,所述接收模块,具体用于接收所述第一客户端通过浏览器发送的所述第一访问请求;所述第一返回包用于在所述浏览器显示目标页面,所述目标页面用于检测XSS漏洞。

[0055] 在一个可选的实现方式中,所述接收模块,具体用于接收所述第一客户端通过测试工具发送的所述第一访问请求;所述测试工具运行于所述第一客户端,且用于根据所述第一返回包检测XSS漏洞。

[0056] 在一个可选的实现方式中,所述代码注入模块,具体用于将所述参考数据中的至少一个字符串类型的数据替换为所述第一攻击代码以得到所述目标数据。

[0057] 在一个可选的实现方式中,所述代码注入模块,还用于在初始数据中注入XSS的第二攻击代码以得到中间数据;所述初始数据包括所述服务端待通过所述CGI向第二客户端发送的数据;所述第二攻击代码与所述第一攻击代码不同;所述CGI,还用于根据所述中间数据构建第二返回包,并通过所述CGI向所述第二客户端发送所述第二返回包,所述第二返回包用于检测XSS漏洞。

[0058] 在一个可选的实现方式中,所述第一标识用于指示所述第一客户端为用于检测第一XSS漏洞的客户端,所述第一标识对应所述第一攻击代码;所述接收单元,还用于接收来自所述第二客户端的第二访问请求,所述第二访问请求用于获取所述初始数据;所述第二访问请求包括第二标识,所述第二标识用于指示所述第二客户端为用于检测第二XSS漏洞的客户端;所述第二标识对应所述第二攻击代码;所述CGI,还用于从所述存储器或所述数据库获取针对所述第二访问请求的所述初始数据。

[0059] 在一个可选的实现方式中,所述服务端还包括:处理单元,用于过滤所述目标数据中会导致所述第一攻击代码对应的脚本执行的内容,和/或,对所述目标数据进行HTML编码。

[0060] 第三方面,本发明实施例提供了另一种服务端,该服务端包括接收器和发送器,还包括:处理器,适于实现一条或多条指令;以及,计算机存储介质,所述计算机存储介质存储有一条或多条指令,所述一条或多条指令适于由所述处理器加载并执行如上述第一方面以及上述第一方面中可选的实现方式的方法。

[0061] 第四方面，本发明实施例提供了一种计算机存储介质，所述计算机存储介质存储有一条或多条指令，所述一条或多条指令适于由处理器加载并执行如上述第一方面以及上述第一方面中可选的实现方式的方法。

附图说明

[0062] 为了更清楚地说明本发明实施例技术方案，下面将对实施例描述中所需要使用的附图作简单地介绍，显而易见地，下面描述中的附图是本发明的一些实施例，对于本领域普通技术人员来讲，在不付出创造性劳动的前提下，还可以根据这些附图获得其他的附图。

[0063] 图1是本发明实施例提供的一种网络架构示意图；

[0064] 图2A为本发明实施例提供的一种手动测试场景示意图；

[0065] 图2B为本发明实施例提供的一种自动测试场景示意图；

[0066] 图3为本发明实施例提供的一种安全漏洞检测方法流程图；

[0067] 图4为本发明实施例提供的另一种安全漏洞检测方法流程图；

[0068] 图5为本发明实施例提供的一种在检测XSS漏洞的过程中服务端和客户端之间的交互流程图；

[0069] 图6为本发明实施例提供的一种浏览器界面示意图；

[0070] 图7A和图7B为本发明实施例提供的一种页面对比示意图；

[0071] 图8为本发明实施例提供的另一种在检测XSS漏洞的过程中服务端和客户端之间的交互流程图；

[0072] 图9为本发明实施例提供的另一种在检测XSS漏洞的过程中服务端和客户端之间的交互流程图；

[0073] 图10本发明实施例还提供的一种服务器结构示意图；

[0074] 图11本发明实施例还提供的另一种服务器结构示意图；

[0075] 图12本发明实施例还提供的一种客户端的结构示意图。

具体实施方式

[0076] 为了使本技术领域的人员更好地理解本发明实施例方案，下面将结合本发明实施例中的附图，对本发明实施例中的技术方案进行清楚地描述，显然，所描述的实施例仅仅是本发明一部分的实施例，而不是全部的实施例。

[0077] 本发明的说明书实施例和权利要求书中的术语“包括”和“具有”以及他们的任何变形，意图在于覆盖不排他的包含，例如，包含了一系列步骤或单元。方法、系统、产品或设备不必限于清楚地列出的那些步骤或单元，而是可包括没有清楚地列出的或对于这些过程、方法、产品或设备固有的其它步骤或单元。

[0078] 本发明实施例提供了安全漏洞检测方法，即检测XSS漏洞的方法，为更清楚的描述本发明的方案。下面先介绍一些与XSS相关的知识。

[0079] XSS攻击:跨站脚本攻击(Cross Site Scripting)，为不和层叠样式表(Cascading Style Sheets,CSS)的缩写混淆，故将跨站脚本攻击缩写为XSS。跨站脚本攻击，是Web程序中常见的漏洞，XSS属于被动式且用于客户端的攻击方式，所以容易被忽略其危害性。其原理是攻击者在网页中嵌入恶意代码，当其它用户浏览该网站时，这段代码会自动执行，从而

达到攻击的目的。XSS并不限于可见的页面输入，还有可能是隐藏表单域、get请求参数等。XSS的本质是：恶意代码未经过滤，与网站正常的代码混在一起；浏览器无法分辨哪些脚本是可信的，导致恶意脚本被执行。可以理解，如果服务端没有将用户输入的文本进行合适的过滤，就贸然插入到HTML中，这很容易造成注入漏洞。攻击者可以利用这些漏洞，构造出恶意的代码指令，进而利用恶意代码危害数据安全。

[0080] XSS存在的根本原因：对URL中的参数，对用户输入提交给Web服务器的内容，没有进行充分的过滤。如果我们能够在Web程序中，对用户提交的URL中的参数，和提交的所有内容，进行充分的过滤，将所有的不合法的参数和输入内容过滤掉，那么就不会导致“在用户的浏览器中执行攻击者自己定制的脚本”。但是，其实充分而完全的过滤，实际上是无法实现的。因为攻击者有各种各样的神奇的，你完全想象不到的方式来绕过服务器端的过滤，最典型的就是对URL和参数进行各种的编码来绕过XSS过滤。

[0081] XSS分类：根据攻击的来源，XSS攻击可分为存储型、反射型和DOM(Document Object Model)型三种。

[0082] 存储型XSS的攻击步骤如下：(1)攻击者将恶意代码提交到目标网站的数据库中；(2)用户打开目标网站时，网站服务端将恶意代码从数据库取出，拼接在HTML中返回给浏览器；(3)用户浏览器接收到响应后解析执行，混在其中的恶意代码也被执行。这种攻击常见于带有用户保存数据的网站功能，如论坛发帖、商品评论、用户私信等。

[0083] 反射型XSS的攻击步骤如下：(1)攻击者构造出特殊的统一资源定位符(Uniform Resource Locator,URL)，其中包含恶意代码；(2)用户打开带有恶意代码的URL时，网站服务器将恶意代码从URL中取出，拼接在HTML中返回给浏览器；(3)用户浏览器接收到响应后解析执行，混在其中的恶意代码也被执行。反射型XSS跟存储型XSS的区别是：存储型XSS的恶意代码存在数据库里，反射型XSS的恶意代码存在URL里。反射型XSS漏洞常见于通过URL传递参数的功能，如网站搜索、跳转等。由于需要用户主动打开恶意的URL才能生效，攻击者往往会结合多种手段诱导用户点击。

[0084] DOM型XSS的攻击步骤如下：(1)攻击者构造出特殊的URL，其中包含恶意代码；(2)用户打开带有恶意代码的URL；(3)用户浏览器接收到响应后解析执行，前端JavaScript取出URL中的恶意代码并执行。DOM型XSS跟前两种XSS的区别：DOM型XSS攻击中，取出和执行恶意代码由浏览器端完成，属于前端JavaScript自身的安全漏洞，而其他两种XSS都属于服务端的安全漏洞。

[0085] XSS攻击的危害：盗取用户储存在用户本地终端上的数据(即Cookie)、破坏页面结构、导航到恶意网站、获取浏览器信息、携带木马、非法转账、强制发送电子邮件、控制受害者机器向其它网站发起攻击、盗取各类用户帐号等。

[0086] XSS攻击的预防：存储型和反射型XSS都是在服务端取出恶意代码后，插入到响应HTML里的，攻击者刻意编写的“数据”被内嵌到“代码”中，被浏览器所执行。预防这两种漏洞，有两种常见做法：1、改成纯前端渲染，把代码和数据分隔开。2、对HTML做充分转义。纯前端渲染的过程：(1)浏览器先加载一个静态HTML，此HTML中不包含任何跟业务相关的数据；(2)浏览器执行HTML中的JavaScript；(3)JavaScript通过Ajax加载业务数据，调用DOM应用程序编程接口(Application Programming Interface,API)更新到页面上。其中，Ajax即“AsynchronousJavaScriptAndXML”(异步JavaScript和XML)，是指一种创建交互式网页应

用的网页开发技术。如果拼接HTML是必要的，就需要采用合适的转义库，对HTML模板各处插入点进行充分的转义。对HTML做充分转义可以包括将不可信数据放入到HTML标签内的时候进行HTML编码、将不可信数据放入HTML属性时进行HTML Attribute编码、将不可信数据作为URL参数值时需要对参数进行URL编码等。上述不可信数据是指用户的输入数据。在任何时候用户的输入都是不可信的，对于不可信数据的输出要进行相应的编码。以上仅列举了几种常见的预防XSS的方式，在实际应用还可以采用其他的方法来预防XSS。

[0087] 在实际应用中，通常会采用一些预防XSS攻击的方式来减少XSS漏洞。然而，在实际应用中，服务端或客户端即便采用了一些预防XSS攻击的措施，可能仍然存在XSS漏洞。为减少服务端或客户端的XSS漏洞，就需要通过测试的方式来找出服务端或客户端存在的XSS漏洞，进而采取相应的策略来防御XSS。另外，在某个网站正式投入使用之前，通常也需要检测XSS漏洞，进而采取相应的预防措施。目前，通常使用通用XSS攻击字符串手动检测XSS漏洞。具体的，手动检测XSS漏洞的方式可以是在网页中的输入框(Textbox)或者其他能输入数据的地方，输入这些测试脚本，看能不能弹出对话框，能弹出的话说明存在XSS漏洞；也可以是在URL中查看有那些变量通过URL把值传给Web服务器，把这些变量的值替换成测试脚本(即替换CGI输入参数值)，然后看测试脚本是否能执行。然而，这种手动检测XSS漏洞的方式仅能检测到从CGI注入恶意代码的XSS漏洞，不能较好的检测存储型XSS漏洞以及其他形式的XSS漏洞。也就是说，这种测试方式不能较好的模拟存储型XSS漏洞以及其他形式的XSS漏洞，即XSS的攻击代码注入率覆盖率低，也就是不能全面的测试防御XSS的能力。另外，这些检测XSS漏洞的方式XSS攻击代码的注入通常需要依赖现有URL规则，欠缺及时性、变化性。因此，需要研究XSS的攻击代码的注入率覆盖率更高的XSS漏洞检测方法。

[0088] 本发明实施例提供的XSS漏洞检测方法主要应用于两种测试场景，一种是手动测试场景，另一种是自动测试场景。图1为本发明实施例提供的一种网络架构示意图。如图1所示，所述网络架构可以包括服务器以及客户端集群；所述客户端集群可以包括多个客户端。如图1所示，客户端集群包括客终端1、客户端2、…、客户端N；如图1所示，客户端1、客终端2、…、客户端N可以分别与服务器进行网络连接，以与服务器进行交互并登陆相应的网页。为更好的理解本方案，本发明实施例在所述客户端集群中选择一个客户端为例来描述客户端与所述服务器之间的数据交互关系。

[0089] 图2A为本发明实施例提供的一种手动测试场景示意图。如图2A所示，客户端通过浏览器向服务端发送访问请求；服务端通过CGI从存储后台(数据库或存储器)获取针对该访问请求的参考数据，通过代码注入模块将XSS的攻击代码注入到该参考数据，根据该参考数据构建第一返回包并通过CGI发送给客户端；客户端的浏览器根据该第一返回包显示相应的页面。在该测试场景中，用户可以根据浏览器显示的页面来确定是否存在XSS漏洞。在该页面包括恶意内容的情况下，可以认为存在XSS漏洞；在该页面未包括恶意内容的情况下，可以认为不存在XSS漏洞。恶意内容是指浏览器执行XSS的攻击代码得到的内容。可以理解，若该页面不包括恶意内容，则表示不存在XSS漏洞，即浏览器和服务端可以有效预防XSS；否则，表示存在XSS漏洞，即浏览器和服务端不能有效预防XSS，需要加强预防XSS的措施。

[0090] 图2B为本发明实施例提供的一种自动测试场景示意图。如图2B所示，客户端上的测试工具向服务端发送访问请求；服务端通过CGI从存储后台(数据库或存储器)获取针对

该访问请求的参考数据，通过代码注入模块将XSS的攻击代码注入到该参考数据，根据该参考数据构建第一返回包并通过CGI发送给客户端；该测试工具对该第一返回包对应的页面源码进行扫描；如果在扫描中发现该页面源码中残留特定代码端(即测试的攻击代码)，则认为存在XSS漏洞，否则，认为不存在XSS漏洞。该测试工具包括扫描工具，该扫描工具用于对该第一返回包对应的页面源码进行扫描。

[0091] 为提高检测XSS漏洞的过程中XSS攻击代码注入的覆盖率，以及检测各种类型的XSS漏洞，本发明实施例中将服务端待通过CGI向客户端发送的数据注入XSS的攻击代码，从而覆盖客户端所有引用CGI层的变量。可以理解，通过这种方式增加了XSS的攻击代码的注入途径，可涵盖CGI流程中所有的外部引入变量，最终提高在XSS漏洞检测中XSS攻击代码的注入覆盖率。下面结合附图介绍服务端执行的操作。

[0092] 图3为本发明实施例提供的一种安全漏洞检测方法流程图。如图3所示，该方法可包括：

[0093] 301、服务端在参考数据中注入跨站脚本攻击XSS的第一攻击代码以得到目标数据。

[0094] 所述参考数据包括所述服务端待通过通用网关接口CGI向第一客户端发送的数据。服务端可以是Web服务器或者其他服务器。该服务器可以为图2A和图2B中的服务端。可以理解，服务端可以在通过CGI向第一客户端发送参考数据之前，在该参考数据中注入XSS的第一攻击代码，从而使得XSS的攻击代码能够覆盖第一客户端所有引用CGI层的变量。XSS的第一攻击代码可以根据测试需要进行设置。例如第一客户端的浏览器运行该第一攻击代码使得该第一客户端弹出某个窗口。又例如第一客户端的浏览器运行该第一攻击代码被导航到恶意网站。该第一攻击代码可以用于检测XSS漏洞的代码，例如导致客户端的标签强制闭合的代码“"></script><script>alert(1)</script><!--”、导致客户端触发资源加载的代码“<img src＝x onerror＝alert(1)>”等，本发明实施例不作限定。

[0095] 所述服务端在参考数据中注入跨站脚本攻击XSS的第一攻击代码以得到目标数据可以是：所述服务端将所述参考数据中的至少一个字符串类型的数据替换为所述第一攻击代码以得到所述目标数据。由于XSS均是将客户端请求获取的数据中的字符串类型的数据替换为攻击代码。因此，通过将参考数据中的字符串类型的数据替换为攻击代码，可以模拟任何类型的XSS，实现简单。在实际测试过程中，服务端可一次只替换其中一段代码；全部功能测试完一遍之后，再换一段代码，重新测试。XSS的攻击代码可以由客户端(扫描工具、或浏览器)决定使用哪几种。可选的，服务端在执行301之前，服务端接收来自客户端的目标指令，所述目标指令用于指示所述服务端将第一攻击代码注入至服务端待通过通用网关接口CGI向第一客户端发送的数据中。可以理解，客户端可以向服务端发送指令以使得服务器注入该指令所指示的攻击代码。这样就可以方便的测试不同的攻击代码。

[0096] 302、服务端根据目标数据构建第一返回包，并通过CGI向第一客户端发送所述第一返回包。

[0097] 该第一返回包用于检测XSS漏洞。该客户端可以是手机、平板电脑、笔记本电脑、台式电脑等可以浏览网页的设备。服务端在参考数据中注入的第一攻击代码作用于业务组装返回包逻辑中。

[0098] XSS防御的一种思路是：对输入(和URL参数)进行过滤，对输出进行编码。也就是对

客户端(即用户)提交的所有内容进行过滤,以及对URL中的参数进行过滤,过滤掉会导致XSS脚本执行的相关内容;然后对动态输出到页面的内容进行HTML编码,使XSS脚本无法在浏览器中执行。虽然对输入过滤可以被绕过,但是也还是会拦截很大一部分的XSS攻击。本发明实施例的XSS漏洞检测方案可以用于检测服务端能否过滤掉会导致XSS脚本执行的相关内容,和/或,合理地对动态输出到页面的内容进行HTML编码,使得XSS脚本无法在浏览器中执行。可以理解,服务端运行有至少一种过滤会导致XSS脚本执行的相关内容的方法,和/或,至少一种对动态输出到页面的内容进行HTML编码的方式。该相关内容可以是“<”、“>”、“&”、回车和空格等。举例来说,过滤“<”的代码可以是replace(str,“<”,“<”)。服务端对待通过CGI向客户端发送的数据注入攻击代码可以是为了检验该服务端这种对动态输出到页面的内容进行HTML编码的方式,和/或,过滤会导致XSS脚本执行的相关内容的策略。可以理解,服务端在执行302之前,可以过滤目标数据中导致XSS脚本执行的相关内容,和/或,在构建返回包之前对动态输出到页面的内容进行HTML编码。

[0099] 举例来说,服务端执行的HTML编码的方式包括“<转成<”、“>转成>”、“&转成&”、“″转成"”、“’转成'”等,服务端在根据目标数据构建返回包之前,可以将该目标数据中的字符串替换为XSS的攻击代码,执行该HTML编码后根据编码后的目标数据构建返回包;客户端解析该返回包,并执行解析该返回包得到的页面源码,从而测试服务端执行的HTML编码的方式是否有效。若客户端显示的页面源码中没有包含完整的XSS的攻击代码,,则服务端执行的HTML编码的方式有效;否则,服务端执行的HTML编码的方式无效。完整的XSS的攻击代码的定义是:XSS的攻击代码没有经过HTML编码,和替换的XSS攻击代码一致。

[0100] 本发明实施例中,服务端将待通过CGI向客户端发送的数据均注入XSS的攻击代码以覆盖客户端所有引用CGI层的变量,能够提高测试XSS漏洞的过程中XSS注入的覆盖率,进而测试各种类型的XSS漏洞。

[0101] 在一个可选的实现方式中,服务端在执行步骤301和步骤302之前,还可以执行如下操作:

[0102] 303、服务端接收来自第一客户端的第一访问请求。

[0103] 该第一访问请求用于获取所述参考数据。可选的,所述第一访问请求包括第一标识,所述第一标识用于指示所述第一客户端为用于检测XSS漏洞的客户端。在实际应用中,服务端可以与一部分客户端(访问请求中包括第一标识的客户端)进行交互来检测XSS漏洞,也可以处理其他客户端的访问请求(不包括该第一标识)来执行其他任务。也就是说,服务端不限于执行XSS漏洞检测任务,还可以同时执行其他任务,例如其他测试任务。可选的,第一标识用于指示所述第一客户端为用于检测第一XSS漏洞的客户端,所述第一标识对应所述第一攻击代码。

[0104] 如图2A所示,服务端接收来自所述第一客户端的访问请求可以是该服务端接收该第一客户端通过浏览器发送的所述访问请求;所述第一返回包用于在所述浏览器显示目标页面,所述目标页面用于检测XSS漏洞。该目标页面可包括浏览器执行XSS的攻击代码得到的内容,例如弹框。在目标页面包括恶意内容的情况下,可以认为存在XSS漏洞;在该目标页面未包括恶意内容的情况下,可以认为不存在XSS漏洞。可以理解,若目标页面不包括恶意内容,则表示不存在XSS漏洞,即浏览器和服务端可以有效预防XSS;否则,表示存在XSS漏

洞,即浏览器和服务端不能有效预防XSS,需要加强预防XSS的措施。

[0105] 所述服务端接收来自第一客户端的访问请求可以是服务端接收所述客户端通过测试工具发送的所述访问请求;所述测试工具运行于第一客户端,且用于根据所述第一返回包检测XSS漏洞。该测试工具可以是运行于第一客户端的专门用于检测XSS漏洞的自动化测试工具。第一客户端通过测试工具发送访问请求可以是该测试工具模拟用户点击页面或链接、页面滚动、键盘输入等常规用户操作(对应于发送访问请求),以便于触发页面交互逻辑,即与服务端进行交互的操作。测试工具可以对第一返回包(即页面源码)进行扫描以查找第一返回包中必须要处理的特定代码段(即XSS的攻击代码);如果在扫描中发现该第一返回包中残留这样的代码,则认为存在XSS漏洞,否则,认为不存在XSS漏洞。该测试工具可以包括一个扫描工具,用于对页面源码进行扫描。该测试工具在检测到特定代码段后,可以记录本次的检测信息,例如检测的时间、点击的链接等;也可以生成一份测试报告,该测试报告用于表示检测到的XSS的情况。该测试报告可包括如下内容中的一项或多项:任务编号,测试开始时间,测试结束时间,测试持续时间;业务信息:使用的帐号信息,接入的服务器信息,覆盖多少页面,执行了哪几种注入代码;有问题的页面信息:执行时间,操作前后录屏,发生问题的页面代码,具体有问题的代码段,发生问题的截图,漏洞安全评级;针对问题的处理:处理结果(人、时间、结论)、创建工单按钮等。

[0106] 304、服务端从存储器或数据库获取针对该第一访问请求的参考数据。

[0107] 该存储器或数据库对应于图2A和图2B中的存储后台。

[0108] 在该实现方式中,服务器根据来自第一客户端的第一访问请求,从存储器或数据库获取相应的数据,操作简单。

[0109] 图3中的第一攻击代码可以是用于测试第一XSS漏洞的攻击代码。该第一攻击代码被第一客户端的浏览器执行后可以是弹出某个窗口,也可以是将浏览器当前显示的页面导航到某个恶意网站,还可以是其他恶意行为。在实际测试过程中,往往需要测试服务端和客户端过滤多种XSS的攻击代码的能力,而不是仅测试服务端和客户端过滤某一种XSS的攻击代码的能力。这样就需要服务端将待通过CGI向客户端发送的数据注入不同的攻击代码。举例来说,服务端将第一攻击代码注入到第一客户端请求访问的数据中,进而测试服务端和该第一客户端能否过滤掉该第一攻击代码;服务端将第二攻击代码注入到第二客户端请求访问的数据中,进而测试服务端和该第二客户端能够过滤掉该第二攻击代码。其中,该第一攻击代码被第一客户端的浏览器执行后可以是弹出某个窗口,该第二攻击代码被第二客户端的浏览器执行后可以是将浏览器当前显示的页面导航到某个恶意网站。可以理解,为更全面的检测XSS漏洞,需要服务端对待通过CGI向客户端发送的数据注入不同的攻击代码。

[0110] 为提高测试效率,本发明实施例中可以同时使用多个客户端来检测不同的攻击代码导致的XSS漏洞。下面以第一客户端和第二客户端为例来介绍如何同时检测多种攻击代码导致的XSS漏洞。图4为本发明实施例提供的另一种安全漏洞检测方法流程图。服务器在执行上述301至304的过程中,还可以执行图4中的漏洞检测方法流程。如图4所示,该方法可包括:

[0111] 401、服务端接收来自第二客户端的第二访问请求。

[0112] 所述第二访问请求包括第二标识,所述第二标识用于指示所述第二客户端为用于检测第二XSS漏洞的客户端;所述第二标识对应所述第二攻击代码。可以理解,若服务端检

测到包括第二标识的访问请求，则向该访问请求所请求的数据中注入第二攻击代码；若服务端检测到包括第一标识的访问请求，则向该访问请求所请求的数据中注入第一攻击代码。

[0113] 402、服务端从存储器或数据库获取针对该第二访问请求的初始数据。

[0114] 403、服务端在初始数据中注入XSS的第二攻击代码以得到中间数据。

[0115] 所述初始数据包括所述服务端待通过所述CGI向第二客户端发送的数据；所述第二攻击代码与所述第一攻击代码不同。403的实现方式与301的实现方式可以相同，也可以不同。

[0116] 404、服务端根据中间数据构建第二返回包，并通过所述CGI向第二客户端发送第二返回包。

[0117] 所述第二返回包用于检测XSS漏洞。404的实现方式与302的实现方式相同。

[0118] 本发明实施例中，服务端根据访问请求携带的标识来对该访问请求所请求的数据注入不同的攻击代码，能够同时检测不同的攻击代码导致的XSS漏洞，测试效率高。

[0119] 下面结合在检测XSS漏洞的过程中服务端和客户端之间的交互流程示意图，来进一步描述检测XSS漏洞的过程。图5为本发明实施例提供的一种在检测XSS漏洞的过程中服务端和客户端之间的交互流程图。图5中检测XSS漏洞的过程对应于图2A中的手动测试场景。如图5所示，服务端和客户端共同完成XSS漏洞检测，具体检测XSS漏洞的过程如下：

[0120] 501、客户端通过浏览器向服务端发送访问请求。

[0121] 该客户端可以是手机、平板电脑、笔记本电脑、台式电脑等可以浏览网页的设备。该浏览器可以是运行于该客户端的浏览器，例如QQ浏览器。客户端通过浏览器向服务器发送访问请求可以用户通过该浏览器的输入框输入网址或搜索信息；也可以是该用户通过点击、动作手势、滑动等输入操作来触发页面交互逻辑，例如点击页面中的链接等。图6为本发明实施例提供的一种浏览器界面示意图。图6为客户端(即手机)显示的浏览器界面。如图6所示，该浏览器界面包括一个输入框(参阅图6中最上方的虚线框)，用户通过该输入框可以输入搜索信息或者网页地址以打开相应的网页；该浏览器界面包括多个链接地址(图6中每个虚线框对应一个链接地址)，用户点击一个链接地址则打开相应的网页。在测试过程中，测试人员通过输入框输入网页地址或搜索信息，以及点击浏览器界面的链接地址的操作即为通过浏览器向服务端发送访问请求。

[0122] 502、服务端从存储后台获取针对访问请求的参考数据。

[0123] 如图5所示，服务器包括CGI、存储后台以及代码注入模块。该存储后台可以是服务端的存储器，也可以是该服务器关联的数据库。代码注入模块，用于将XSS的攻击代码注入参考数据以得到目标数据。

[0124] 503、服务端在参考数据中注入XSS的攻击代码以得到目标数据。

[0125] 服务端在参考数据中注入XSS的攻击代码以得到目标数据可以是服务端调用代码注入模块在该参考数据中注入XSS的攻击代码以得到该目标数据。

[0126] 504、服务端根据目标数据构建返回包。

[0127] 505、服务端通过CGI向客户端发送返回包。

[0128] 506、客户端执行该返回包对应的页面源码以显示目标页面。

[0129] 客户端执行该返回包对应的页面源码以显示目标页面可以是客户端解析该返回

包以得到该页面源码，执行该页面源码以显示该目标页面。该目标页面可包括浏览器执行XSS的攻击代码得到的内容，例如弹框。在目标页面包括恶意内容的情况下，可以认为存在XSS漏洞；在该目标页面未包括恶意内容的情况下，可以认为不存在XSS漏洞。可以理解，若目标页面不包括恶意内容，则表示不存在XSS漏洞，即浏览器和服务端可以有效预防XSS；否则，表示存在XSS漏洞，即浏览器和服务端不能有效预防XSS，需要加强预防XSS的措施。图7A和图7B为本发明实施例提供的一种页面对比示意图。图7A为服务端未对客户端请求的数据注入XSS的攻击代码时，客户端解析来自该服务端的返回包显示的页面(即正常页面)。图7B为服务端对客户端请求的数据注入XSS的攻击代码时，客户端解析来自该服务端的返回包显示的页面。对比图7A和图7B可以看出，服务端对客户端请求的数据注入XSS的攻击代码，客户端显示的页面出现攻击脚本，即图7B中虚线框中的内容。

[0130] 本发明实施例中，客户端通过浏览器向服务器发送访问请求，并通过该浏览器显示的页面来确定是否存在XSS漏洞；能够模拟实际的应用场景，可以方便、直观的检测XSS漏洞。

[0131] 图8为本发明实施例提供的另一种在检测XSS漏洞的过程中服务端和客户端之间的交互流程图。如图8所示，服务端和客户端共同完成XSS漏洞检测，具体检测XSS漏洞的过程如下：

[0132] 801、客户端通过浏览器向服务端发送访问请求。

[0133] 步骤801的实现方式可以与步骤501的实现方式相同。

[0134] 802、CGI向代码注入模块发送攻击代码获取请求。

[0135] 如图8所示，服务器包括CGI、存储后台以及代码注入模块。该存储后台可以是服务端的存储器，也可以是该服务器关联的数据库。该攻击代码获取请求用于请求第一攻击代码。该访问请求可以携带有该客户端的标识。可选的，CGI在执行步骤802之前，根据该访问请求携带的客户端的标识，确定待获取第一攻击代码；生成攻击代码获取请求，该攻击代码获取请求用于请求该第一攻击代码。CGI可存储有客户端的标识与攻击代码的对应关系，根据该对应关系可以确定任一客户端的标识对应的攻击代码，即待获取的攻击代码。可选的，CGI在执行步骤802之前，生成攻击代码获取请求，该攻击代码获取请求用于请求第一攻击代码。该第一攻击代码为服务端当前待测试的攻击代码。

[0136] 803、代码注入模块获取第一攻击代码。

[0137] 代码注入模块关联的存储器存储有至少一种攻击代码。可选的，代码注入模块从其关联的存储器中获取该攻击代码获取请求所请求的第一攻击代码。

[0138] 804、代码注入模块向CGI发送第一攻击代码。

[0139] 805、CGI向代码注入模块发送数据获取请求。

[0140] 该数据获取请求携带有所述第一攻击代码，且用于请求参考数据。

[0141] 806、代码注入模块从存储后台获取参考数据。

[0142] 该参考数据为所述数据获取请求所请求的数据。代码注入模块通过解析数据获取请求，可确定待从该存储后台获取的数据以及待获取数据的定义。

[0143] 807、代码注入模块从存储后台获取参考定义。

[0144] 该参考定义用于指示参考数据中各字符的类型。应理解，根据该参考定义可准确地确定参考数据中各字符串类型的数据。

[0145] 808、代码注入模块根据参考定义，将参考数据中字符串类型的数据替换为第一攻击代码以得到目标数据。

[0146] 可选的，代码注入模块可根据该参考定义，将该参考数据中未被标注的各字符串类型的数据替换为第一攻击代码以得到目标数据。也就是说，该参考数据中被标注的字符串类型的数据不被替换为该第一攻击代码。为保证向客户端发送的参考数据满足业务要求，服务端会标注该参考数据中的一些字符串类型的数据，以便代码注入模块就不会用该第一攻击代码替换这些数据。

[0147] 809、CGI根据目标数据构建返回包。

[0148] 810、CGI向客户端发送返回包。

[0149] 811、客户端执行该返回包对应的页面源码以显示目标页面。

[0150] 步骤808至步骤811的实现可参考图5中的步骤504至步骤506。

[0151] 本申请实施例中，CGI通过代码注入模块从存储器或数据库获取参考数据，相当于在CGI和存储后台之间多了一层中间代理层，可以提高数据获取速度，并且能够避免CGI直接访问存储后台造成的其他问题。

[0152] 图9为本发明实施例提供的另一种在检测XSS漏洞的过程中服务端和客户端之间的交互流程图。图9中检测XSS漏洞的过程对应于图2B中的自动测试场景。如图9所示，服务端和客户端共同完成XSS漏洞检测，具体检测XSS漏洞的过程如下：

[0153] 901、客户端通过扫描工具向服务端发送访问请求。

[0154] 901与图5中的501类似。该扫描工具可以是运行于客户端的专门用于检测XSS漏洞的自动化测试工具的一部分。客户端通过扫描工具发送访问请求可以是该扫描工具模拟用户点击页面或链接、页面滚动、键盘输入等常规用户操作，以便于触发页面交互逻辑，即与服务端进行交互的操作。

[0155] 902、服务端从存储后台获取针对访问请求的参考数据。

[0156] 如图9所示，服务器包括CGI、存储后台以及代码注入模块。该存储后台可以是服务端的存储器，也可以是该服务器关联的数据库。代码注入模块，用于将XSS的攻击代码注入参考数据以得到目标数据。

[0157] 903、服务端在参考数据中注入XSS的攻击代码以得到目标数据。

[0158] 904、服务端执行XSS防御策略以处理目标数据。

[0159] 服务端执行XSS防御策略以处理目标数据可以是：所述服务端过滤所述目标数据中会导致XSS的攻击代码对应的脚本执行的内容，和/或，对所述目标数据进行HTML编码。服务端还可以执行其他XSS防御策略来处理目标数据，本发明实施例不作限定。可以理解，通过图9中的方法可以测试服务端当前执行的XSS防御策略是否可以有效防御XSS。若服务端当前执行的XSS防御策略不能有效的防御XSS，则需要进一步完善XSS防御策略，这是测试的一个目的。

[0160] 905、服务端根据处理后的目标数据构建返回包。

[0161] 906、服务通通过CGI向客户端发送返回包。

[0162] 905至906可以依次对应图5中的504至505。

[0163] 907、客户端通过扫描工具对解析返回包得到的页面源码进行扫描，以确定XSS漏洞的存在情况。

[0164] 客户端通过扫描工具对解析返回包得到的页面源码进行扫描，以确定XSS漏洞的存在情况可以是该扫描工具解析该返回包得到页面源码，并对该页面源码进行扫描以查找特定代码段(即XSS的攻击代码)；如果在扫描中发现该页面源码上残留这样的代码，则认为存在XSS漏洞，否则，认为不存在XSS漏洞。

[0165] 可选的，该测试工具在检测到特定代码段(即XSS的攻击代码)后，可以记录本次的检测信息，例如检测的时间、点击的链接等；也可以生成一份测试报告，该测试报告用于说明检测到的XSS的情况。可以理解，测试工具可以自动向服务器发送第一访问请求以触发页面交互逻辑，并对交互后的页面源码进行扫描以检测XSS漏洞，测试效率高、操作简单。

[0166] 可选的，客户端还可以通过浏览器执行该返回包对应的页面源码以显示目标页面。这样可以测试这些攻击代码是否被浏览器执行，以及这些攻击代码被执行后生成的攻击脚本。

[0167] 本发明实施例中，服务器在客户端请求的数据中注入攻击代码，并向该客户端发送返回包，以便于运行于该客户端上的测试工具根据该返回包来检测XSS漏洞，可以有效提高测试效率。

[0168] 基于上述安全漏洞检测方法实施例的描述，本发明实施例还公开了一种服务器，所述服务器可以运行有一个计算机程序(包括程序代码)。该服务器可以执行图3至图6以及图9中服务端执行的方法。请参见图10，所述服务器可以运行如下模块：

[0169] 代码注入模块1001，用于在参考数据中注入跨站脚本攻击XSS的第一攻击代码以得到目标数据；所述参考数据包括所述服务端待通过通用网关接口CGI向第一客户端发送的数据；

[0170] CGI1002，用于根据所述目标数据构建第一返回包，并向所述第一客户端发送所述第一返回包，所述第一返回包用于检测XSS漏洞。

[0171] 在一种实施方式中，所述服务端还包括：接收模块1003，用于接收来自所述第一客户端的第一访问请求，所述第一访问请求用于获取所述参考数据；

[0172] CGI1002，还用于从存储器或数据库获取针对所述第一访问请求的所述参考数据。

[0173] 再一种实施方式中，所述第一访问请求包括第一标识，所述第一标识用于指示所述第一客户端为用于检测XSS漏洞的客户端。

[0174] 再一种实施方式中，接收模块1003，具体用于接收所述第一客户端通过浏览器发送的所述第一访问请求；所述第一返回包用于在所述浏览器显示目标页面，所述目标页面用于检测XSS漏洞。

[0175] 再一种实施方式中，接收模块1003，具体用于接收所述第一客户端通过测试工具发送的所述第一访问请求；所述测试工具运行于所述第一客户端，且用于根据所述第一返回包检测XSS漏洞。

[0176] 再一种实施方式中，代码注入模块1001，具体用于将所述参考数据中的至少一个字符串类型的数据替换为所述第一攻击代码以得到所述目标数据。

[0177] 再一种实施方式中，代码注入模块1001，还用于在初始数据中注入XSS的第二攻击代码以得到中间数据；所述初始数据包括所述服务端待通过所述CGI向第二客户端发送的数据；所述第二攻击代码与所述第一攻击代码不同；

[0178] CGI1002，还用于根据所述中间数据构建第二返回包，并通过所述CGI向所述第二

客户端发送所述第二返回包,所述第二返回包用于检测XSS漏洞。

[0179] 再一种实施方式中,所述第一标识用于指示所述第一客户端为用于检测第一漏洞的客户端,所述第一标识对应所述第一攻击代码;

[0180] 所述接收单元,还用于接收来自所述第二客户端的第二访问请求,所述第二访问请求用于获取所述初始数据;所述第二访问请求包括第二标识,所述第二标识用于指示所述第二客户端为用于检测第二XSS漏洞的客户端;所述第二标识对应所述第二攻击代码;

[0181] CGI1002,还用于从所述存储器或所述数据库获取针对所述第二访问请求的所述初始数据。

[0182] 根据本发明实施例,图3至图6以及图8至图9中服务端执行的方法所涉及的各个步骤均可以是由图10所示的服务器中的各个单元来执行的。例如,图3中所示的301由图10中所示的代码注入模块1001来执行,302由图10中的CGI1002来执行;又如,图4中所示的401可以由图10中所示的接收摸1003来执行,403可以由图10中的CGI1002来执行,403可以由代码注入模块1002来执行,404可以由CGI1002来执行。

[0183] 根据本发明实施例,图10所示的服务器中的各个单元可以分别或全部合并为一个或若干个另外的单元来构成,或者其中的某个(些)单元还可以再拆分为功能上更小的多个单元来构成,这可以实现同样的操作,而不影响本发明的实施例的技术效果的实现。上述单元是基于逻辑功能划分的,在实际应用中,一个单元的功能也可以由多个单元来实现,或者多个单元的功能由一个单元实现。在本发明的其它实施例中,基于服务端也可以包括其它单元,在实际应用中,这些功能也可以由其它单元协助实现,并且可以由多个单元协作实现。

[0184] 根据本发明的另一个实施例,可以通过在包括中央处理单元(CPU)、随机存取存储介质(RAM)、只读存储介质(ROM)等处理元件和存储元件的例如计算机的通用计算设备上运行能够执行如图3至图6以及图8中所示的相应方法所涉及的各步骤的计算机程序(包括程序代码),来构造如图10中所示的服务器,以及来实现本发明实施例的安全漏洞检测方法。所述计算机程序可以记载于例如计算机可读记录介质上,并通过计算机可读记录介质装载于上述服务器中,并在其中运行。

[0185] 基于上述方法实施例以及装置实施例的描述,本发明实施例还提供了一种服务器结构示意图,该服务器1100可因配置或性能不同而产生比较大的差异,可以包括一个或一个以上中央处理器(central processing units,CPU)1122(例如,一个或一个以上处理器)和存储器1132,一个或一个以上存储应用程序1142或数据1144的存储介质1130(例如一个或一个以上海量存储设备)。其中,存储器1132和存储介质1130可以是短暂存储或持久存储。存储在存储介质1130的程序可以包括一个或一个以上模块(图示没标出),每个模块可以包括对服务器中的一系列指令操作。更进一步地,中央处理器1122可以设置为与存储介质1130通信,在服务器1100上执行存储介质1130中的一系列指令操作。服务器1100可以为本发明提供的服务端。

[0186] 服务器1100还可以包括一个或一个以上电源1126,一个或一个以上有线或无线网络接口1150,一个或一个以上输入输出接口1158,和/或,一个或一个以上操作系统1141,例如Windows ServerTM,Mac OS XTM,UnixTM,LinuxTM,FreeBSDTM等等。

[0187] 上述实施例中由服务端所执行的步骤可以基于该图11所示的服务器结构。具体

的,中央处理器1122可实现代码注入模块1001、CGI1002的功能;输入输出接口1158可实现接收模块1003的功能。

[0188] 进一步地,请参见图12,是本发明实施例提供的一种客户端的结构示意图。如图12所示,所述客户端1210对应于上述实施例中的客户端,所述客户端1210可以包括:至少一个处理器1201,例如CPU,至少一个网络接口1204,用户接口1203,存储器1205,至少一个通信总线1202。其中,通信总线1202用于实现这些组件之间的连接通信。其中,用户接口1203可以包括显示屏(Display)、键盘(Keyboard),可选的,用户接口1203还可以包括标准的有线接口、无线接口。网络接口1204可选地可以包括标准的有线接口、无线接口(如WI-FI接口)。存储器1205可以是高速RAM存储器,也可以是非不稳定的存储器(non-volatile memory),例如至少一个磁盘存储器。存储器1205可选地还可以是至少一个位于远离前述处理器1201的存储装置。如图12所示,作为一种计算机存储介质的存储器1205中可以包括操作系统、网络通信模块、用户接口模块以及设备控制应用程序。

[0189] 在图12所示的客户端1200中,网络接口1204主要用于连接客户端和服务器;而用户接口1203主要用于为用户提供输入的接口;而处理器1201可以用于调用存储器1205中存储的设备控制应用程序,以实现:向服务端发送访问请求;接收来自服务端的返回包;解析该返回包得到页面源码并执行该页面源码以显示目标页面,或者,解析该返回包得到页面源码并扫描该页面源码以检测XSS的攻击代码。

[0190] 应当理解,客户端1200可以执行前述实施例中客户端执行的操作,例如图5和图8中客户端执行的操作。

[0191] 此外,这里需要指出的是:本发明实施例还提供了一种计算机存储介质,且所述计算机存储介质中存储有前文提及的服务端所执行的计算机程序,且所述计算机程序包括程序指令,当所述处理器执行所述程序指令时,能够执行前文图3至图6或图8所对应实施例中对所述安全漏洞检测方法的描述,因此,这里将不再进行赘述。另外,对采用相同方法的有益效果描述,也不再进行赘述。对于本发明所涉及的计算机存储介质实施例中未披露的技术细节,请参照本发明方法实施例的描述。

[0192] 本领域普通技术人员可以理解实现上述实施例方法中的全部或部分流程,是可以通过计算机程序来指令相关的硬件来完成,所述的程序可存储于一计算机可读取存储介质中,该程序在执行时,可包括如上述各方法的实施例的流程。其中,所述的存储介质可为磁碟、光盘、只读存储记忆体(Read-Only Memory,ROM)或随机存储记忆体(Random Access Memory,RAM)等。

[0193] 以上所揭露的仅为本发明较佳实施例而已,当然不能以此来限定本发明之权利范围,因此依本发明权利要求所作的等同变化,仍属本发明所涵盖的范围。

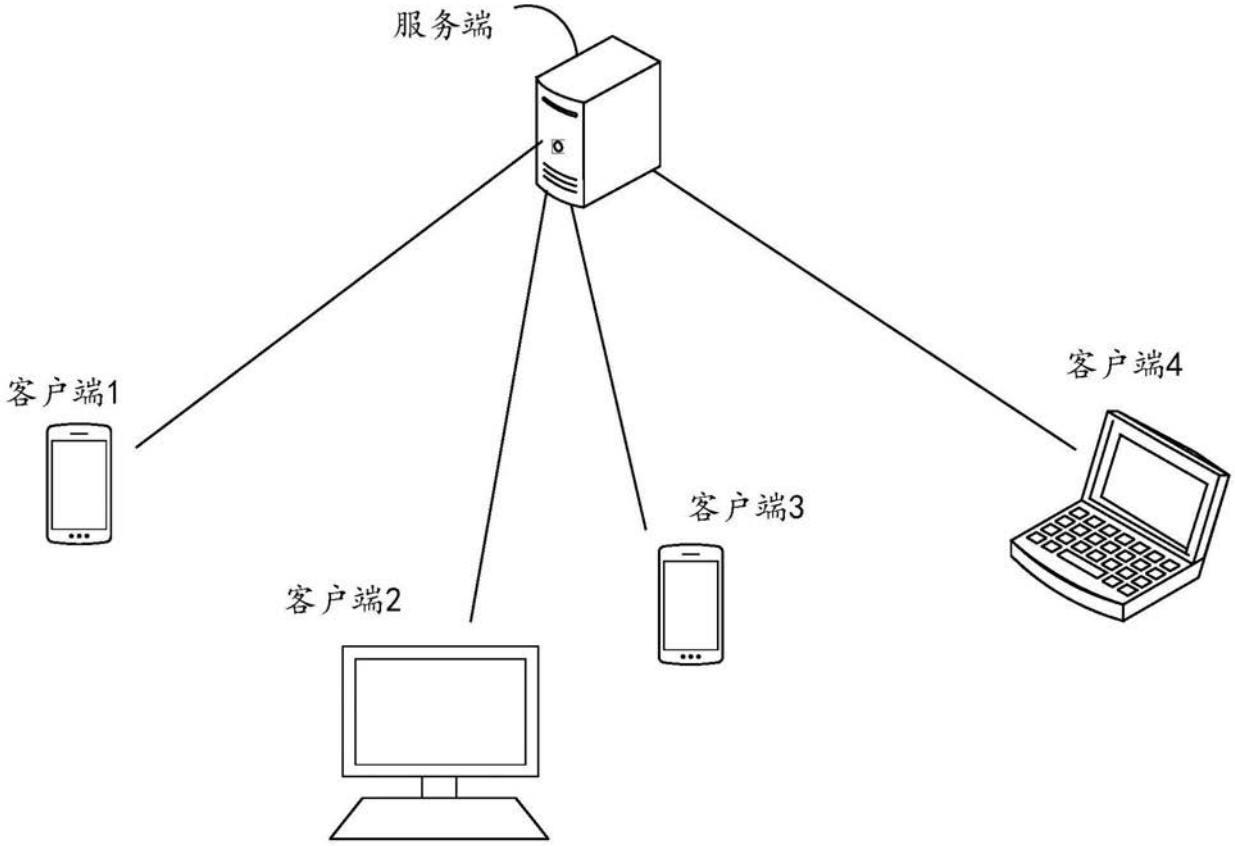


图1

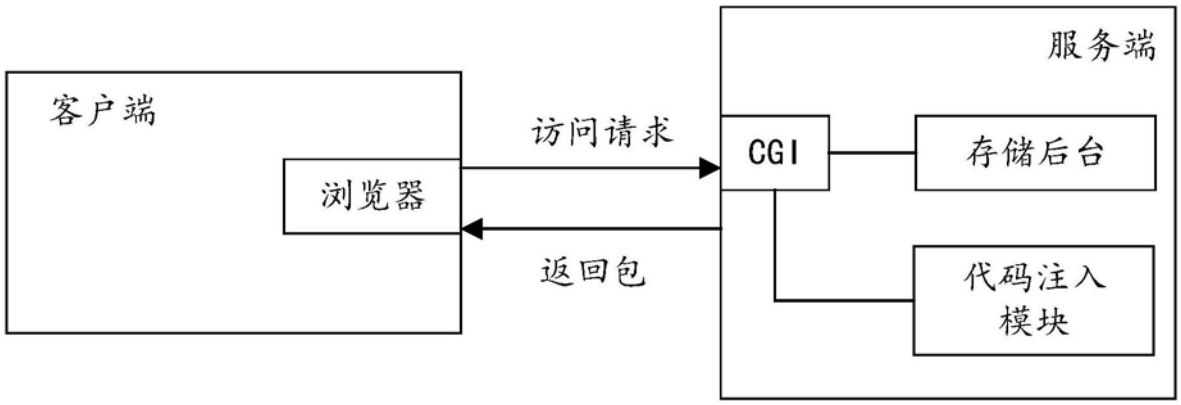


图2A

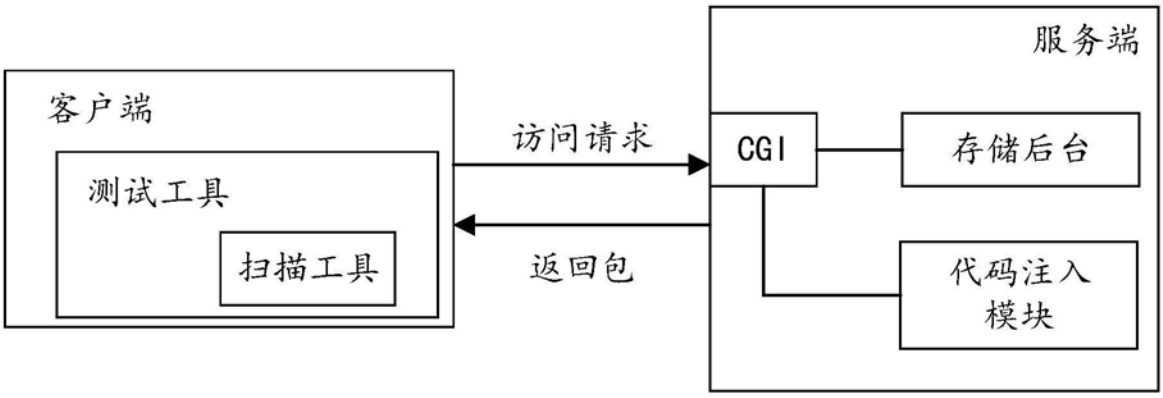

服务端
客户端
测试工具
扫描工具
访问请求
返回包
CGI
存储后台
代码注入
模块

图2B

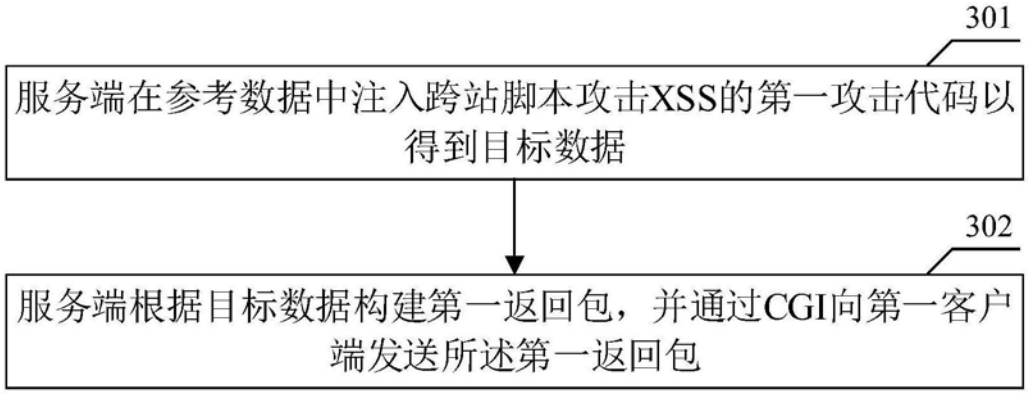

301
服务端在参考数据中注入跨站脚本攻击XSS的第一攻击代码以得到目标数据
302
服务端根据目标数据构建第一返回包，并通过CGI向第一客户端发送所述第一返回包

图3

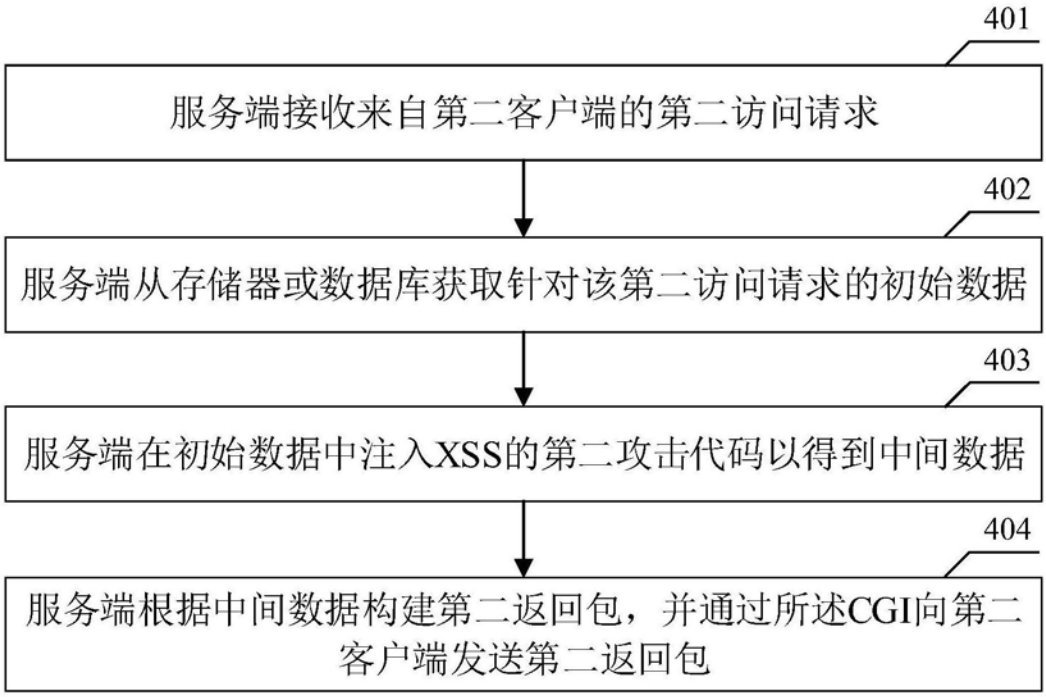

401
服务端接收来自第二客户端的第二访问请求
402
服务端从存储器或数据库获取针对该第二访问请求的初始数据
403
服务端在初始数据中注入XSS的第二攻击代码以得到中间数据
404
服务端根据中间数据构建第二返回包，并通过所述CGI向第二客户端发送第二返回包

图4

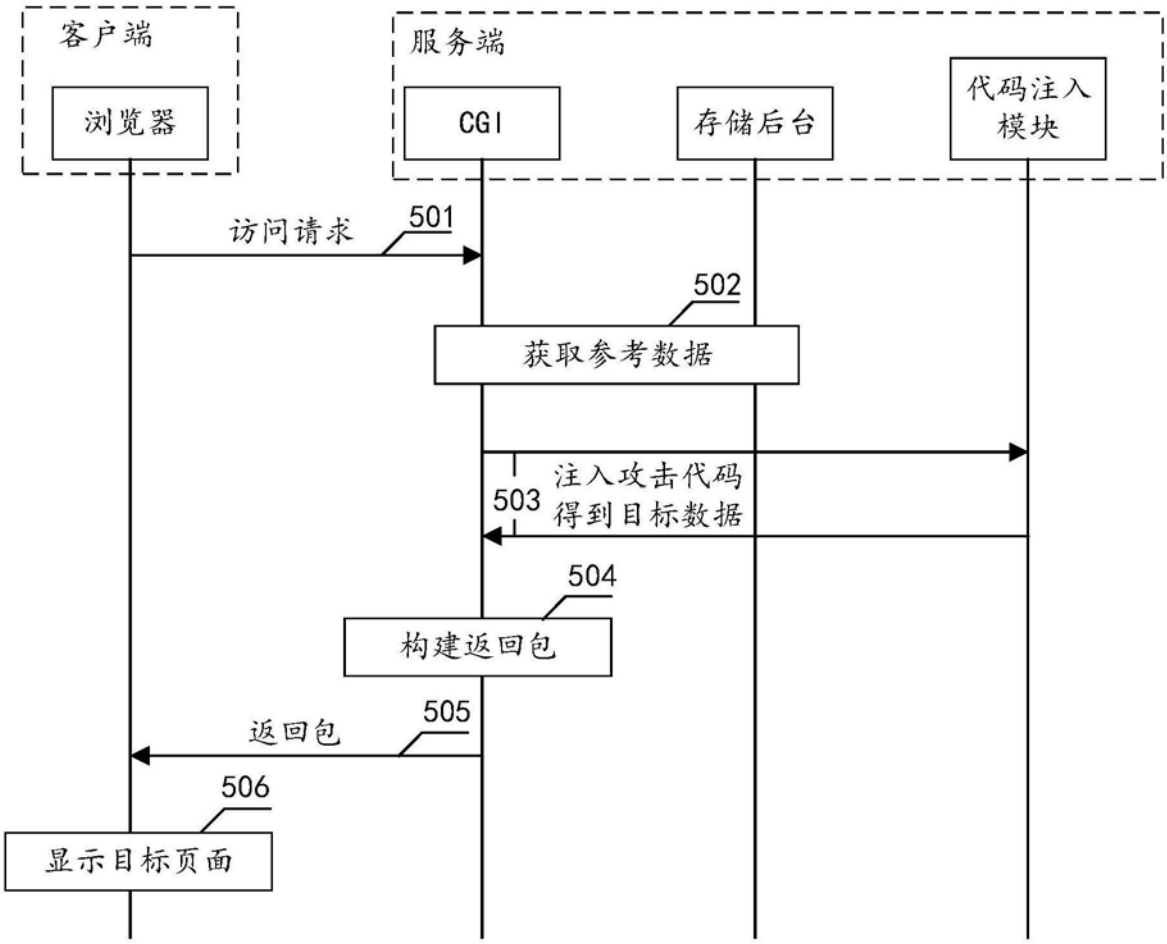

客户端
浏览器
服务端
CGI
存储后台
代码注入模块
访问请求
501
502
获取参考数据
注入攻击代码
503
得到目标数据
504
构建返回包
505
返回包
506
显示目标页面

图5


中国移动
中国联通
66% 晚上9:39
25° 坪山 阴
空气 优
06月02日
农历四月廿九
端午节热门短途游
直播交友
小说
漫画
想听FM
游戏
网址导航
全球热点
立即获取
应用大全
下载领Q币
全部
更多小工具
推荐
视频
深圳
娱乐
体育
明星
@全体党员，这12字要牢记！
置顶 人民日报客户端
竖版微视频丨出发吧，少年追梦人
置顶 央视网
电动车"替代品"终于有了，一公里不到1毛钱，既合法又安全
Pragma industries
汽车小涛 203评
首页
视频
我的
文件
多窗口

图6


A科技公司
头像
小明 邀请你加入：
A科技公司
创建人 小明
加入后，可以通过企业通讯录找到其他同事
通过企业微信提供的应用高效率完成工作
加入企业
加入后，可在微信中收发企业消息

图7A


'"></script><script>alert(1)</script><"'
...
头像
'"></script><script>...邀请你加入:
'"></script><script>...
创建人
'"></script><script>alert(1)</script><"'
加入后，可以通过企业通讯录找到其他同事
通过企业微信提供的应用高效率完成工作
加入企业
加入后，可在微信中收发企业消息

图7B

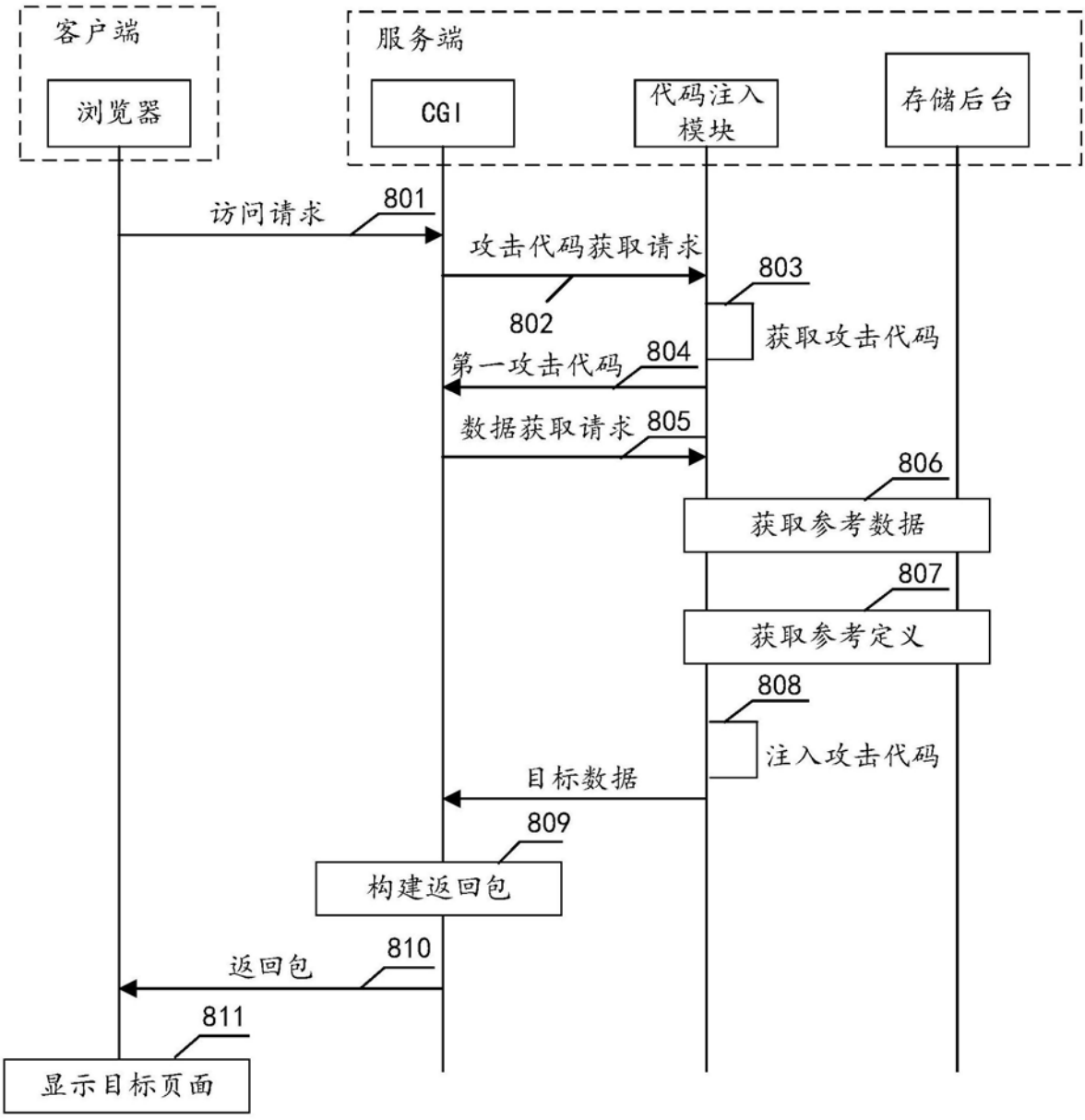

客户端
浏览器
服务端
CGI
代码注入模块
存储后台
访问请求
801
攻击代码获取请求
802
803
获取攻击代码
第一攻击代码
804
数据获取请求
805
806
获取参考数据
807
获取参考定义
808
注入攻击代码
目标数据
809
构建返回包
返回包
810
811
显示目标页面

图8

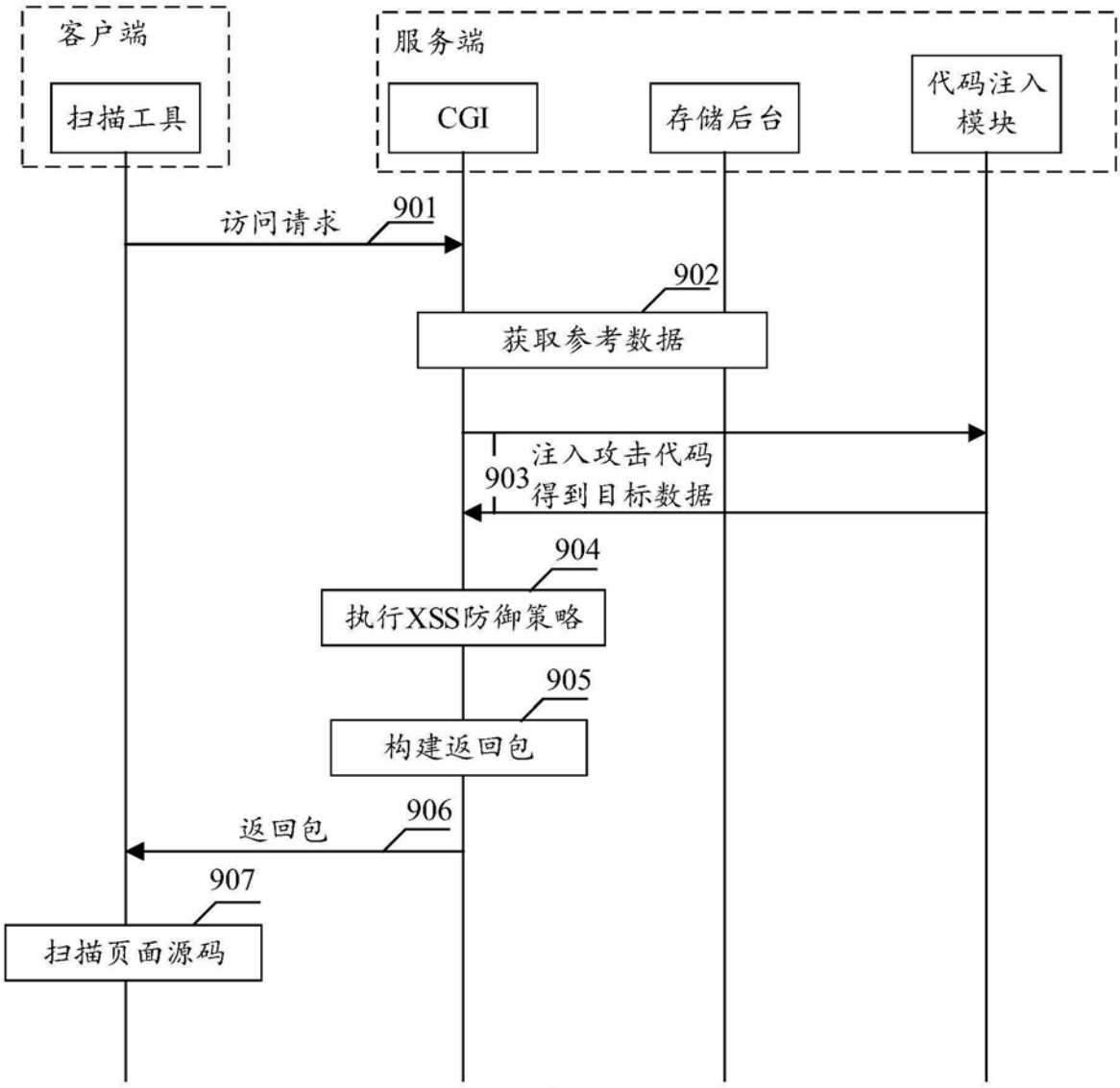

客户端
扫描工具
服务端
CGI
存储后台
代码注入模块
访问请求
901
902
获取参考数据
注入攻击代码
903
得到目标数据
904
执行XSS防御策略
905
构建返回包
906
返回包
907
扫描页面源码

图9

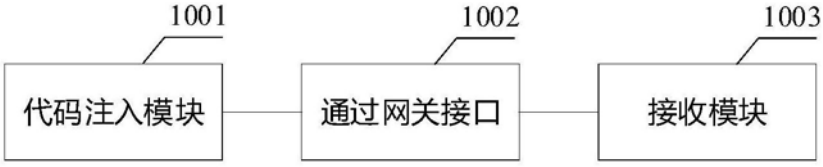

1001
代码注入模块
1002
通过网关接口
1003
接收模块

图10

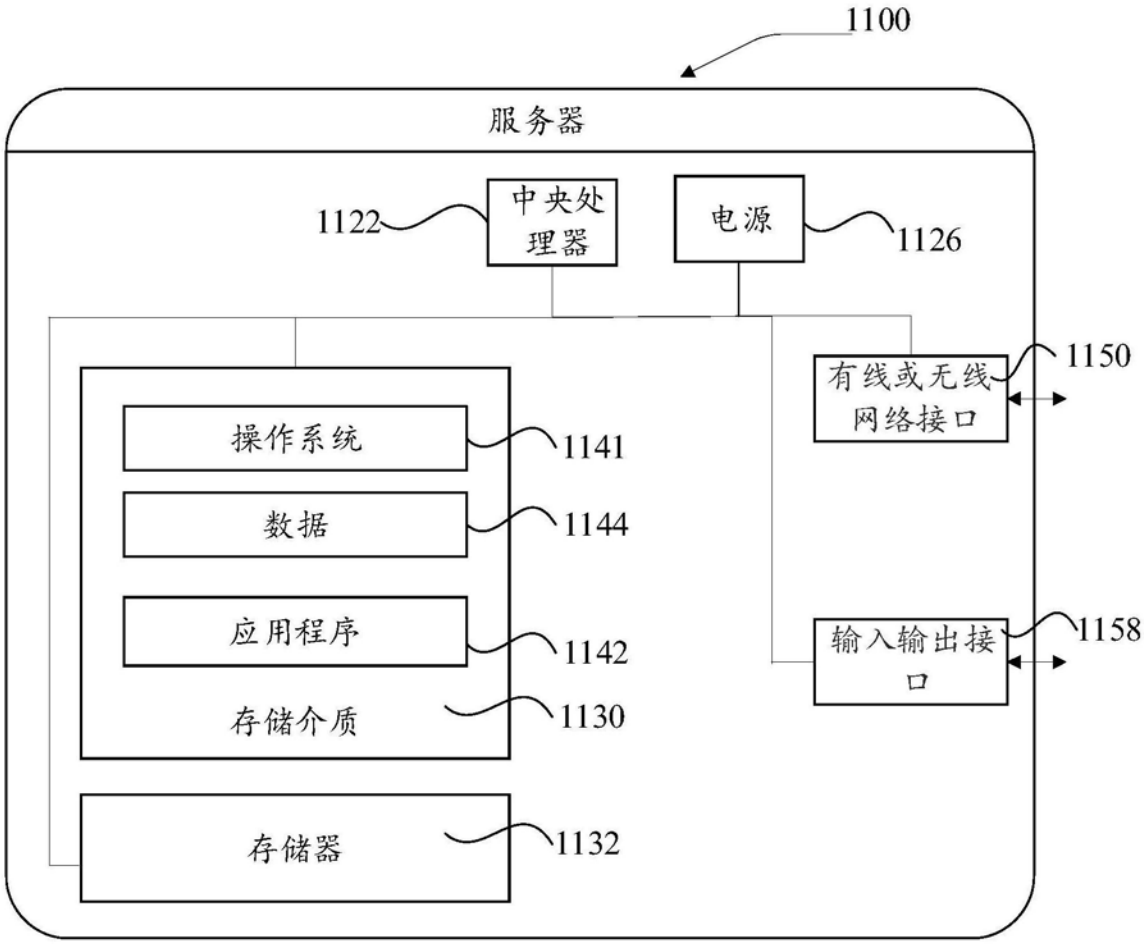

1100
服务器
1122
中央处理器
电源
1126
有线或无线网络接口
1150
操作系统
1141
数据
1144
应用程序
1142
存储介质
1130
输入输出接口
1158
存储器
1132

图11

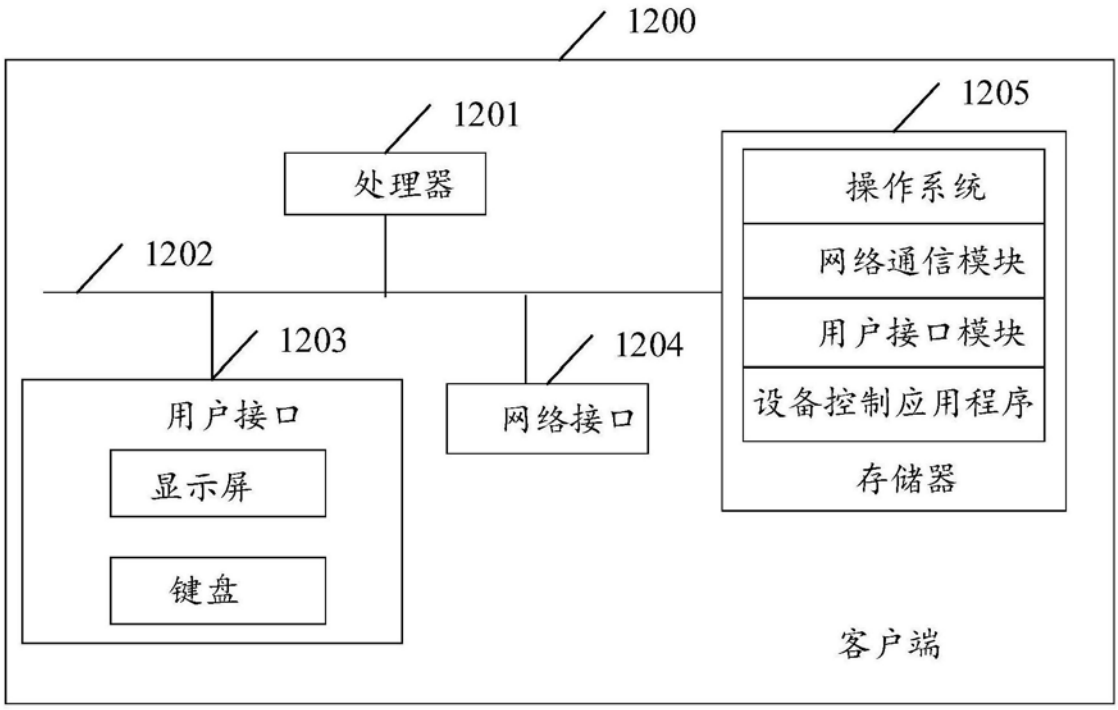

1200
1201
处理器
1202
1203
用户接口
显示屏
键盘
1204
网络接口
1205
操作系统
网络通信模块
用户接口模块
设备控制应用程序
存储器
客户端

图12