(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 105871850 A
(43)申请公布日 2016.08.17

(21)申请号 201610206481.0

(22)申请日 2016.04.05

(71)申请人 携程计算机技术(上海)有限公司
地址 200335 上海市长宁区福泉路99号携程网络技术大楼

(72)发明人 陈剑 李巍

(74)专利代理机构 上海弼兴律师事务所 31283
代理人 薛琦 王聪

(51)Int.Cl.
H04L 29/06(2006.01)

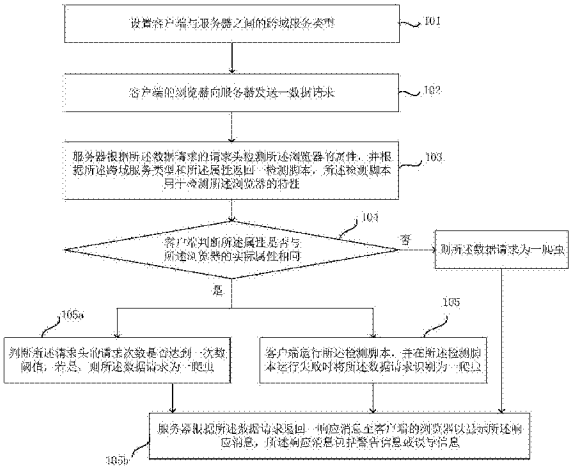
权利要求书2页 说明书6页 附图2页

(54)发明名称

爬虫检测方法和系统

(57)摘要

本发明公开了爬虫检测方法和系统，该方法包括：设置客户端与服务器之间的跨域服务类型；客户端的浏览器向服务器发送一数据请求；服务器根据所述数据请求的请求头检测所述浏览器的属性，并根据所述跨域服务类型和所述属性返回一检测脚本，所述检测脚本用于检测所述浏览器的特性；客户端判断所述属性是否与所述浏览器的实际属性相同，若是，客户端运行所述检测脚本，并在所述检测脚本运行失败时将所述数据请求识别为一爬虫；若否，则所述数据请求为一爬虫。本发明能够从浏览器的角度出发，能够检测用任意种类的浏览器来获取服务器端的数据的爬虫，保护了服务器不受非法请求的攻击，节省了服务器资源，降低了网络带宽消耗。



CN 105871850 A

1.一种爬虫检测方法,其特征在于,包括:

S_1、设置客户端与服务器之间的跨域服务类型;

S_2、客户端的浏览器向服务器发送一数据请求;

S_3、服务器根据所述数据请求的请求头检测所述浏览器的属性,并根据所述跨域服务类型和所述属性返回一检测脚本,所述检测脚本用于检测所述浏览器的特性;

S_4、客户端判断所述属性是否与所述浏览器的实际属性相同,若是,进入步骤S_5,若否,则所述数据请求为一爬虫;

S_5、客户端运行所述检测脚本,并在所述检测脚本运行失败时将所述数据请求识别为一爬虫。

2.如权利要求1所述的爬虫检测方法,其特征在于,所述服务器内存储有一脚本库,所述脚本库包括若干个浏览器的属性和与每一浏览器的属性对应的至少一检测脚本,步骤S_3中根据所述跨域服务类型和所述属性返回一检测脚本为:

S_{31}、服务器根据所述跨域服务类型和所述属性查询所述脚本库以获得与所述属性对应的检测脚本,并在获得的检测脚本中随机抽取一检测脚本以发送至所述客户端。

3.如权利要求1所述的爬虫检测方法,其特征在于,步骤S_4中在判断结果为是时还执行以下步骤:

S_{5a}、判断所述请求头的请求次数是否达到一次数阈值,若是,则所述数据请求为一爬虫。

4.如权利要求1所述的爬虫检测方法,其特征在于,所述请求头包括客户端的浏览器类型和客户端的浏览器版本,以及包括客户端的硬件平台、客户端的操作系统和客户端的用户偏好中的一项或多项,步骤S_3中服务器根据所述数据请求的请求头检测所述浏览器的属性为:

S_{32}、服务器通过NodeJS的方式检测所述浏览器的属性。

5.如权利要求1-4中任意一项所述的爬虫检测方法,其特征在于,在检测到所述数据请求为一爬虫时还执行以下步骤:

S_{5b}、服务器根据所述数据请求返回一响应消息至客户端的浏览器以显示所述响应消息,所述响应消息包括警告信息或误导信息。

6.如权利要求1-4中任意一项所述的爬虫检测方法,其特征在于,所述跨域服务类型包括JSONP跨域服务,所述浏览器的属性包括浏览器的类别和浏览器的版本;

和/或,步骤S_3还对所述检测脚本执行压缩混淆操作后将压缩混淆的检测脚本返回至客户端。

7.一种爬虫检测系统,其特征在于,包括一客户端和一服务器,所述客户端包括一设置模块、一请求模块、一第一判断模块和一运行模块,所述服务器包括一检测模块;

所述设置模块用于设置客户端与服务器之间的跨域服务类型;

所述请求模块用于通过浏览器向服务器发送一数据请求;

所述检测模块用于根据所述数据请求的请求头检测所述浏览器的属性,并根据所述跨域服务类型和所述属性返回一检测脚本,所述检测脚本用于检测所述浏览器的特性;

所述第一判断模块用于判断所述属性是否与所述浏览器的实际属性相同,若是,调用所述运行模块,若否,则所述数据请求为一爬虫;

所述运行模块用于运行所述检测脚本，并在所述检测脚本运行失败时将所述数据请求识别为一爬虫。

8.如权利要求7所述的爬虫检测系统，其特征在于，所述服务器内存储有一脚本库，所述脚本库包括若干个浏览器的属性和与每一浏览器的属性对应的至少一检测脚本，所述检测模块用于根据所述跨域服务类型和所述属性返回一检测脚本为：

所述检测模块用于根据所述跨域服务类型和所述属性查询所述脚本库以获得与所述属性对应的检测脚本，并在获得的检测脚本中随机抽取一检测脚本以发送至所述客户端。

9.如权利要求7所述的爬虫检测系统，其特征在于，所述服务器还包括一第二判断模块，所述第二判断模块用于判断所述请求头的请求次数是否达到一次数阈值，若是，则所述数据请求为一爬虫。

10.如权利要求7-9中任意一项所述的爬虫检测系统，其特征在于，所述服务器还包括一响应模块，所述响应模块用于在检测到所述数据请求为一爬虫时根据所述数据请求返回一响应消息至客户端的浏览器以显示所述响应消息，所述响应消息包括警告信息或误导信息。

爬虫检测方法和系统

技术领域

[0001] 本发明涉及互联网中的爬虫检测方法和系统。

背景技术

[0002] 爬虫是一种自动获取网页内容的程序，是搜索引擎的重要组成部分。随着互联网的发展，互联网上的爬虫量日益增加，由于爬虫会伪造用户行为，不断地访问服务器以获取信息，导致爬虫会严重拖慢服务器的响应速度，还会浪费网络带宽，甚至会非法盗取信息，威胁信息财产安全，如爬虫对金融类应用的攻击。而现有技术中反爬虫的方法通常是对爬虫所在的IP(Internet Protocol，互联网协议)地址进行封锁，但这样的封锁行为难以防范分散的多个IP地址的爬虫，现有技术中还通常将服务器端返回的数据进行加密，使得爬虫所在的客户端无法破解返回的数据，但仍然存在一些爬虫开发者通过分析页面的代码来破解该加密算法，从而获得想要的数据。可见，现有技术中的反爬虫的方法仍然无法多方面地检测爬虫。

发明内容

[0003] 本发明要解决的技术问题是为了克服现有技术中反爬虫的手段单一、且难以更全面更彻底地检测爬虫的缺陷，提供一种爬虫检测方法和系统。

[0004] 本发明是通过下述技术方案解决上述技术问题的：

[0005] 一种爬虫检测方法，其特点在于，包括：

[0006] S_1、设置客户端与服务器之间的跨域服务类型；

[0007] S_2、客户端的浏览器向服务器发送一数据请求；

[0008] S_3、服务器根据所述数据请求的请求头检测所述浏览器的属性，并根据所述跨域服务类型和所述属性返回一检测脚本，所述检测脚本用于检测所述浏览器的特性；

[0009] S_4、客户端判断所述属性是否与所述浏览器的实际属性相同，若是，进入步骤S_5，若否，则所述数据请求为一爬虫；

[0010] S_5、客户端运行所述检测脚本，并在所述检测脚本运行失败时将所述数据请求识别为一爬虫。

[0011] 本发明通过设置跨域服务类型，使得客户端能够请求不同域的数据，而服务器端能够在响应的信息中携带不同的代码，通过将服务器端检测的浏览器的属性与浏览器的实际属性进行对比，能够检测出伪造了请求头的爬虫，而通过运行服务器端反馈的检测脚本，能够检测出使用浏览器内核模拟器等接近于真实浏览器的特性的浏览器进行访问数据的爬虫。并且，由于服务器端反馈了检测脚本，因此对于爬虫开发者而言，则无法通过分析页面代码的方式来破解服务器端返回的加密的响应消息。而对于分散了IP地址的爬虫，则也无法躲避本申请从浏览器的属性和浏览器本身的特性出发所执行的双重检测拦截，可见，本申请的检测方法从浏览器的角度出发，能够更加全面地、彻底地检测网络内的爬虫。

[0012] 较佳地，所述服务器内存储有一脚本库，所述脚本库包括若干个浏览器的属性和

与每一浏览器的属性对应的至少一检测脚本，步骤S_3中根据所述跨域服务类型和所述属性返回一检测脚本为：

[0013] S_{31}、服务器根据所述跨域服务类型和所述属性查询所述脚本库以获得与所述属性对应的检测脚本，并在获得的检测脚本中随机抽取一检测脚本以发送至所述客户端。

[0014] 其中，浏览器的特性包括浏览器的类型和浏览器的版本所支持的功能和具有的性能，比如浏览器的动画播放功能、浏览器的安全性是否较高、浏览器的速度是否较高、浏览器是否能够兼容某些网站内容、标签浏览、内置RSS(一种用来分发和汇集网页内容的扩展性标识语音格式)支持、多会话恢复、网站的缩略图、网页翻译、立体搜索、颜色配置文件支持、网页更新提醒、阅读模式等等。可以理解，检测脚本更多的是针对浏览器的实际属性和检测到的浏览器的属性所形成的不同浏览器之间的功能差异所执行的检测，甚至是针对某个浏览器本身具有的bug(漏洞)所执行的检测。

[0015] 较佳地，步骤S_4中在判断结果为是时还执行以下步骤：

[0016] S_{5a}、判断所述请求头的请求次数是否达到一次数阈值，若是，则所述数据请求为一爬虫。

[0017] 通过对请求次数的判断，能够进一步针对分散IP地址的爬虫进行检测。

[0018] 较佳地，所述请求头包括客户端的浏览器类型和客户端的浏览器版本，以及包括客户端的硬件平台、客户端的操作系统和客户端的用户偏好中的一项或多项，步骤S_3中服务器根据所述数据请求的请求头检测所述浏览器的属性为：

[0019] S_{32}、服务器通过NodeJS(是一个基于Chrome V8引擎的JavaScript运行环境，JavaScript是一种直译式脚本语言，V8是一个由丹麦Google开发的开源JavaScript引擎，用于Google Chrome浏览器中)的方式检测所述浏览器的属性。

[0020] 较佳地，在检测到所述数据请求为一爬虫时还执行以下步骤：

[0021] S_{5b}、服务器根据所述数据请求返回一响应消息至客户端的浏览器以显示所述响应消息，所述响应消息包括警告信息或误导信息。

[0022] 也即，在检测到爬虫时，发出警告以提示非法用户请求获得服务器的数据的行为，或者直接向非法用户返回虚假的数据信息。

[0023] 较佳地，所述跨域服务类型包括JSONP(是JSON with Padding的略称，它是一个非官方的协议，它允许在服务器端集成Script tags(脚本标签)返回至客户端，通过JavaScript callback(回调函数)的形式实现跨域访问)跨域服务，和/或，所述浏览器的属性包括浏览器的类别和浏览器的版本；

[0024] 和/或，步骤S_3还对所述检测脚本执行压缩混淆操作后将压缩混淆的检测脚本返回至客户端。

[0025] 本发明还提供一种爬虫检测系统，其特点在于，包括一客户端和一服务器，所述客户端包括一设置模块、一请求模块、一第一判断模块和一运行模块，所述服务器包括一检测模块；

[0026] 所述设置模块用于设置客户端与服务器之间的跨域服务类型；

[0027] 所述请求模块用于通过浏览器向服务器发送一数据请求；

[0028] 所述检测模块用于根据所述数据请求的请求头检测所述浏览器的属性，并根据所述跨域服务类型和所述属性返回一检测脚本，所述检测脚本用于检测所述浏览器的特性；

[0029] 所述第一判断模块用于判断所述属性是否与所述浏览器的实际属性相同，若是，调用所述运行模块，若否，则所述数据请求为一爬虫；

[0030] 所述运行模块用于运行所述检测脚本，并在所述检测脚本运行失败时将所述数据请求识别为一爬虫。

[0031] 较佳地，所述服务器内存储有一脚本库，所述脚本库包括若干个浏览器的属性和与每一浏览器的属性对应的至少一检测脚本，所述检测模块用于根据所述跨域服务类型和所述属性返回一检测脚本为：

[0032] 所述检测模块用于根据所述跨域服务类型和所述属性查询所述脚本库以获得与所述属性对应的检测脚本，并在获得的检测脚本中随机抽取一检测脚本以发送至所述客户端。

[0033] 较佳地，所述服务器还包括一第二判断模块，所述第二判断模块用于判断所述请求头的请求次数是否达到一次数阈值，若是，则所述数据请求为一爬虫。

[0034] 较佳地，所述服务器还包括一响应模块，所述响应模块用于在检测到所述数据请求为一爬虫时根据所述数据请求返回一响应消息至客户端的浏览器以显示所述响应消息，所述响应消息包括警告信息或误导信息。

[0035] 本发明的积极进步效果在于：本发明能够从浏览器的角度出发，通过对浏览器的属性和浏览器的特性进行检测，能够检测用任意种类的浏览器，包括伪造请求头的浏览器和模拟的浏览器来获取服务器端的数据的爬虫，且通过携带检测脚本，使得爬虫开发者无法通过分析页面代码来获得服务器端返回的加密的响应消息，而对于分散了IP地址的爬虫，也无法躲避本申请从浏览器的属性和浏览器本身的特性出发所执行的双重检测拦截，并且，通过对请求次数的判断，能够进一步加强对分散IP地址的爬虫进行检测。保护了服务器不受非法请求的攻击，节省了服务器资源，降低了网络带宽消耗。

附图说明

[0036] 图1为本发明实施例1的爬虫检测方法流程图。

[0037] 图2为本发明实施例2的爬虫检测系统的结构示意图。

具体实施方式

[0038] 下面通过实施例的方式进一步说明本发明，但并不因此将本发明限制在所述的实施例范围之中。

[0039] 实施例1

[0040] 本实施例提供一种爬虫检测方法，本实施例的应用场景可为：一非法用户使用一台电脑，并通过浏览器IE8访问网站A的酒店价格数据和价格的排列方式数据，如图1所示，图1中的虚线表示两个设备之间进行网络连接，包括：

[0041] 步骤101、设置客户端与服务器之间的跨域服务类型；

[0042] 跨域是指，由于浏览器同源策略，凡是发送请求url(Uniform Resoure Locator，统一资源定位符)的协议、域名、端口三者之间任意一与当前页面地址不同即为跨域。

[0043] 所述跨域服务类型包括JSONP跨域服务，在不产生冲突的情况下，也可设置为CORS(Cross-Origin Resource Sharing跨域资源共享，是一种允许当前域(domain)的资源被其

他域的脚本请求访问的机制)跨域服务。

[0044] 步骤102、客户端的浏览器向服务器发送一数据请求；

[0045] 该数据请求则为向网站A的服务器访问网站A的酒店价格数据和价格的排列方式的请求，由于该数据请求的载体是建立在客户端和浏览器的基础上的，因此该数据请求内必然携带有客户端和浏览器的信息，该信息通过请求头User-Agent的方式体现，所述请求头包括客户端的浏览器类型和客户端的浏览器版本，以及包括客户端的硬件平台、客户端的操作系统和客户端的用户偏好中的一项或多项。

[0046] 步骤103、服务器根据所述数据请求的请求头检测所述浏览器的属性，并根据所述跨域服务类型和所述属性返回一检测脚本，所述检测脚本用于检测所述浏览器的特性；

[0047] 具体地，服务器通过NodeJS的方式检测所述浏览器的属性，所述属性包括浏览器的类别和浏览器的版本，所述服务器内存储有一脚本库，所述脚本库包括若干个浏览器的属性和与每一浏览器的属性对应的至少一检测脚本，服务器根据所述跨域服务类型和所述属性查询所述脚本库以获得与所述属性对应的检测脚本，并在获得的检测脚本中随机抽取一检测脚本以发送至所述客户端。由于是从脚本库里随机抽取的检测脚本，因此爬虫开发者是无法从浏览器的页面代码层面上来破解服务器响应的数据的。

[0048] 其中，还可对返回的检测脚本通过JavaScript压缩混淆工具对检测脚本做压缩混淆操作，以进一步增加爬虫开发者的破解难度。

[0049] 当爬虫用户频繁更改请求头User-Agent的内容，来伪装成不同的客户端时，如伪装成浏览器IE10或浏览器Chrome时，服务器端检测到的则是伪装了的浏览器版本和浏览器类别。

[0050] 步骤104、客户端判断所述属性是否与所述浏览器的实际属性相同，若是，进入步骤105，若否，则所述数据请求为一爬虫；

[0051] 客户端判断服务器端检测到的属性与客户端实际使用的浏览器的属性是否相同，由于爬虫用户更改了请求头User-Agent的内容，因此不论爬虫用户如何变换请求头，始终能够被本申请的检测方法检测出。

[0052] 步骤105、客户端运行所述检测脚本，并在所述检测脚本运行失败时将所述数据请求识别为一爬虫。

[0053] 而当爬虫用户不采用更改请求头User-Agent的内容的方式来伪装成不同的客户端，而只是采用与真实浏览器类似的运行速度较快的浏览器内核模拟器，如phantom模拟器，以访问服务器的数据时，服务器会根据检测到的浏览器的属性所对应的浏览器的特性，反馈一检测脚本来检测该浏览器的特性，而这样的特性检测必然是只有在真实的浏览器和真实的用户访问行为的情况下才能运行成功，而在一些浏览器内核模拟器上或者一些伪造了请求头User-Agent的情况下运行失败。比如，服务器返回的检测脚本用于检测浏览器是否存在某个bug，针对实际的浏览器而言，该浏览器本身是存在该bug的，但客户端的浏览器运行该检测脚本时，反而不会出现该bug，因此可断定访问用户为一爬虫。再如，服务器返回的检测脚本用于检测浏览器是否具有网页翻译的功能，针对实际的浏览器而言，该浏览器本身是具有网页翻译功能的，但客户端的浏览器运行该检测脚本时，执行网页翻译功能失败，因此也可断定访问用户为一爬虫。该检测脚本可通过JavaScript实现。

[0054] 此外，步骤104中在判断结果为是时还执行以下步骤：

[0055] 步骤105a、判断所述请求头的请求次数是否达到一次数阈值，若是，则所述数据请求为一爬虫。

[0056] 也即，当爬虫用户不更换请求头User-Agent的内容，但仍然使用真实浏览器而非浏览器内核模拟器时，由于减少了伪造浏览器的变种手段，这样必然使得单个用户的请求头User-Agent的访问率突出，在这种情况是再根据访问量的判断是很容易检测爬虫并封锁爬虫的IP地址的。

[0057] 而在检测到所述数据请求为一爬虫时还执行以下步骤：

[0058] 步骤105b、服务器根据所述数据请求返回一响应消息至客户端的浏览器以显示所述响应消息，所述响应消息包括警告信息或误导信息。

[0059] 可以理解，当真实用户使用真实浏览器访问服务器的数据时，且在访问量可接受的情况下，该浏览器是经得起本实施例采用的多重检测手段的，而只有在通过本实施例的多重检测手段后，服务器才会将对应于数据请求的真实的响应数据开放给客户端以做显示，至于真实的响应数据与检测脚本是处于同一数据包中还是不同的数据包中，本实施例不作限定，只要本实施例能够反馈检测脚本和真实的响应数据即可，而当检测到爬虫时，服务器也会根据实际情况发出警告信息以提示非法用户的请求行为，或者直接向非法用户返回虚假的数据信息，甚至是返回一个与数据请求完全无关的数据页面。

[0060] 可见，本实施例能够从浏览器的角度出发，能够检测用任意种类的浏览器，包括伪造请求头的浏览器和模拟的浏览器来获取服务器端的数据的爬虫，且通过携带检测脚本，使得爬虫开发者无法通过分析页面代码来获得服务器端返回的加密的响应消息，而对于分散了IP地址的爬虫，也无法躲避本申请从浏览器的属性和浏览器本身的特性出发所执行的双重检测拦截，并且，通过对请求次数的判断，能够进一步加强对分散IP地址的爬虫进行检测。保护了服务器不受非法请求的攻击，节省了服务器资源，降低了网络带宽消耗，能够防止XSS(Cross Site Scripting，跨站脚本攻击)、SQL(Structured Query Language，结构化查询语言)等攻击，保护了网络安全。能够更加全面地、彻底地检测网络内的爬虫。

[0061] 实施例2

[0062] 本实施例提供一种爬虫检测系统，如图2所示，包括一客户端1和一服务器2，所述客户端1包括一设置模块11、一请求模块12、一第一判断模块13和一运行模块14，所述服务器2包括一检测模块21；

[0063] 所述设置模块11用于设置客户端与服务器之间的跨域服务类型；

[0064] 所述请求模块12用于通过浏览器向服务器发送一数据请求；

[0065] 所述检测模块21用于根据所述数据请求的请求头检测所述浏览器的属性，并根据所述跨域服务类型和所述属性返回一检测脚本，所述检测脚本用于检测所述浏览器的特性；

[0066] 所述第一判断模块13用于判断所述属性是否与所述浏览器的实际属性相同，若是，调用所述运行模块，若否，则所述数据请求为一爬虫；

[0067] 所述运行模块14用于运行所述检测脚本，并在所述检测脚本运行失败时将所述数据请求识别为一爬虫。

[0068] 所述服务器2内存储有一脚本库，所述脚本库包括若干个浏览器的属性和与每一浏览器的属性对应的至少一检测脚本，所述检测模块用于根据所述跨域服务类型和所述属

性返回一检测脚本为：

[0069] 所述检测模块21用于根据所述跨域服务类型和所述属性查询所述脚本库以获得与所述属性对应的检测脚本，并在获得的检测脚本中随机抽取一检测脚本以发送至所述客户端。

[0070] 所述服务器还包括一第二判断模块22，所述第二判断模块22用于判断所述请求头的请求次数是否达到一次数阈值，若是，则所述数据请求为一爬虫。

[0071] 所述服务器还包括一响应模块23，所述响应模块23用于在检测到所述数据请求为一爬虫时根据所述数据请求返回一响应消息至客户端的浏览器以显示所述响应消息，所述响应消息包括警告信息或误导信息。

[0072] 本实施例的爬虫检测系统可以采用实施例1的方法进行工作，该系统主要在客户端上做验证，并在客户端验证通过后才开放服务器端返回的真实的响应数据，还能对爬虫用户执行多重检测，能够获得与实施例1相同的技术效果。

[0073] 虽然以上描述了本发明的具体实施方式，但是本领域的技术人员应当理解，这些仅是举例说明，本发明的保护范围是由所附权利要求书限定的。本领域的技术人员在不背离本发明的原理和实质的前提下，可以对这些实施方式做出多种变更或修改，但这些变更和修改均落入本发明的保护范围。

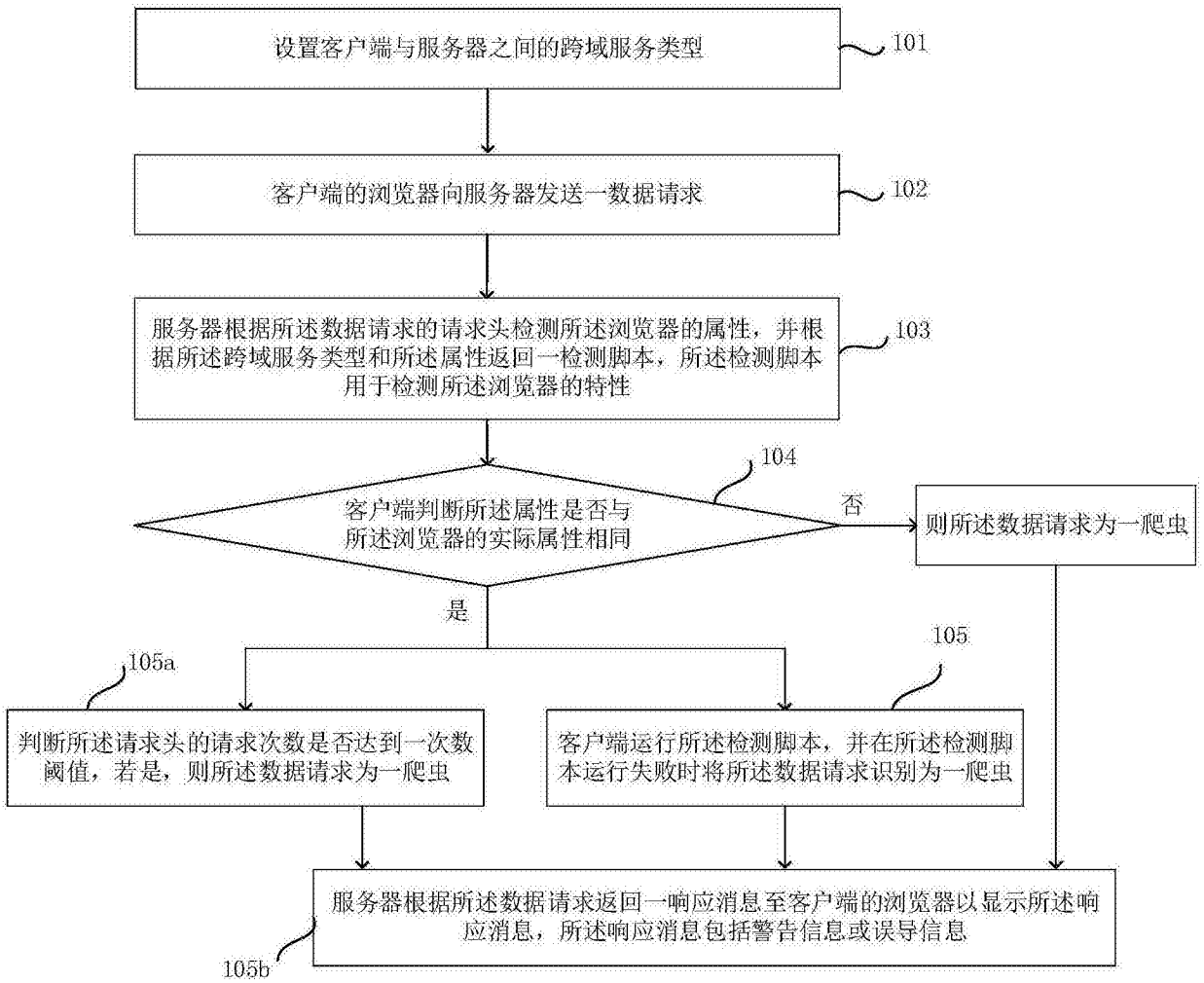

设置客户端与服务器之间的跨域服务类型
101
客户端的浏览器向服务器发送一数据请求
102
服务器根据所述数据请求的请求头检测所述浏览器的属性，并根据所述跨域服务类型和所述属性返回一检测脚本，所述检测脚本用于检测所述浏览器的特性
103
客户端判断所述属性是否与所述浏览器的实际属性相同
104
否
则所述数据请求为一爬虫
是
105a
105
判断所述请求头的请求次数是否达到一次数阈值，若是，则所述数据请求为一爬虫
客户端运行所述检测脚本，并在所述检测脚本运行失败时将所述数据请求识别为一爬虫
服务器根据所述数据请求返回一响应消息至客户端的浏览器以显示所述响应消息，所述响应消息包括警告信息或误导信息
105b

图1

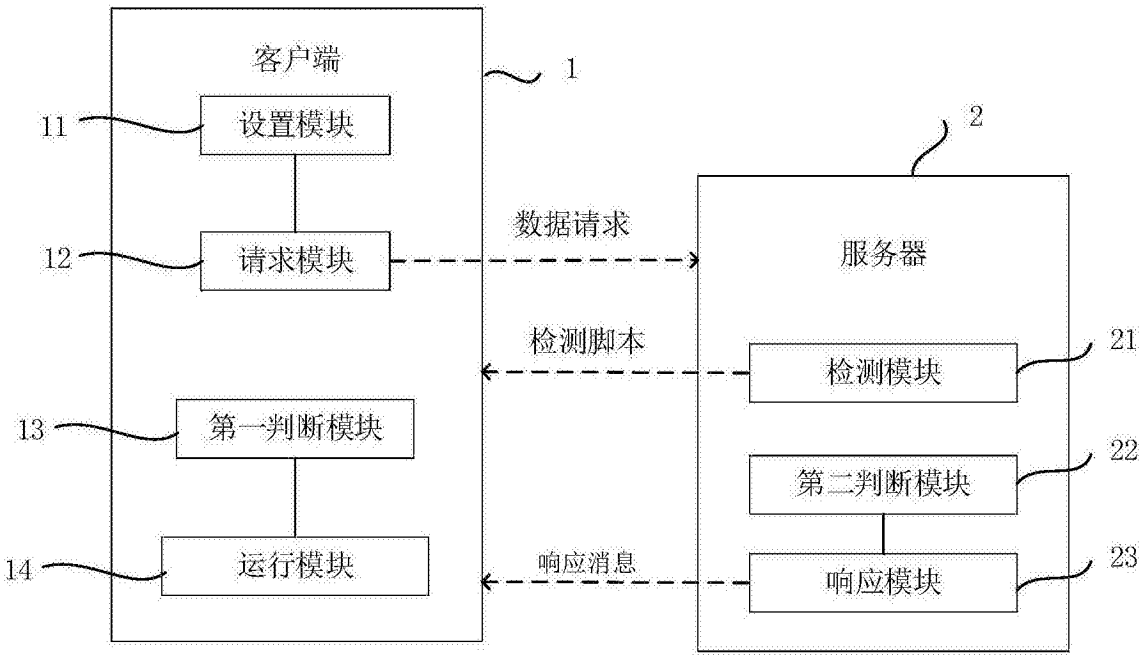

客户端
1
11
设置模块
12
请求模块
数据请求
2
服务器
检测脚本
21
检测模块
13
第一判断模块
22
第二判断模块
14
运行模块
响应消息
23
响应模块

图2