ALIBABA CLOUD

# Alibaba Cloud

## 负载均衡

## 健康检查

文档版本：20220311

阿里云

# 法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。

2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。

3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。

4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的“现状”、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。

5. 阿里云网站上所有内容，包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含“阿里云”、“Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。

6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

# 通用约定

| 格式 | 说明 | 样例 |
| --- | --- | --- |
| 危险 | 该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。 | 危险<br>重置操作将丢失用户配置数据。 |
| 警告 | 该类警示信息可能会导致系统重大变更甚至故障，或者导致人身伤害等结果。 | 警告<br>重启操作将导致业务中断，恢复业务时间约十分钟。 |
| 注意 | 用于警示信息、补充说明等，是用户必须了解的内容。 | 注意<br>权重设置为0，该服务器不会再接受新请求。 |
| 说明 | 用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。 | 说明<br>您也可以通过按Ctrl+A选中全部文件。 |
| > | 多级菜单递进。 | 单击**设置**> **网络**> **设置网络类型**。 |
| **粗体** | 表示按键、菜单、页面名称等UI元素。 | 在**结果确认**页面，单击**确定**。 |
| Courier字体 | 命令或代码。 | 执行 `cd /d C:/window` 命令，进入Windows系统文件夹。 |
| *斜体* | 表示参数、变量。 | `bae log list --instanceid`<br>*Instance_ID* |
| [] 或者 [a\|b] | 表示可选项，至多选择一个。 | `ipconfig [-all\|-t]` |
| {} 或者 {a\|b} | 表示必选项，至多选择一个。 | `switch {active\|stand}` |

# 目录

# 1.健康检查概述

负载均衡通过健康检查来判断后端服务器（ECS实例）的业务可用性。健康检查机制提高了前端业务整体可用性，避免了后端ECS异常对总体服务的影响。

开启健康检查功能后，当后端某台ECS健康检查出现异常时，负载均衡会自动将新的请求分发到其它健康检查正常的ECS上；而当该ECS恢复正常运行时，负载均衡会将其自动恢复到负载均衡服务中。

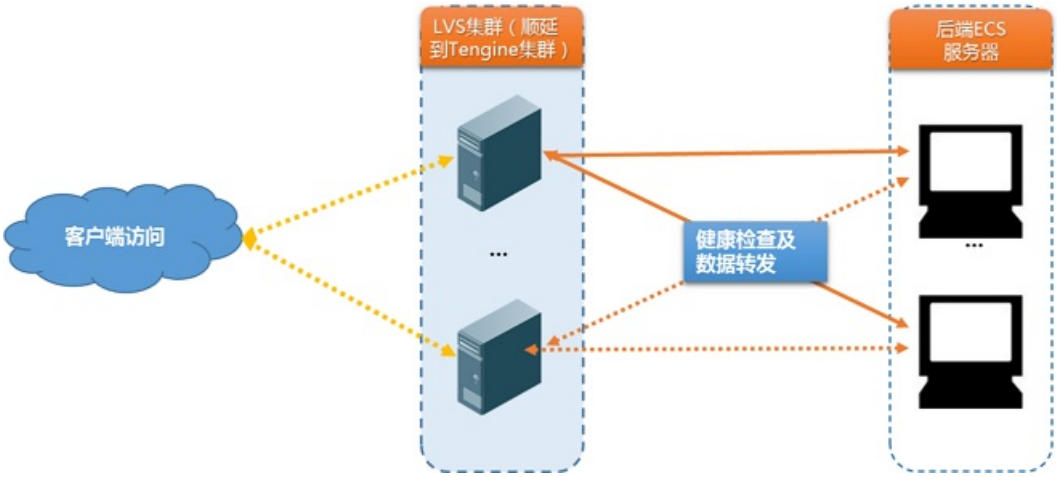
如果您的业务对负载敏感性高，高频率的健康检查探测可能会对正常业务访问造成影响。您可以结合业务情况，通过降低健康检查频率、增大健康检查间隔、七层检查修改为四层检查等方式，来降低对业务的影响。但为了保障业务的持续可用，不建议关闭健康检查。

## 健康检查过程

负载均衡采用集群部署。LVS集群或Tengine集群内的相关节点服务器同时承载了数据转发和健康检查职责。

LVS集群内不同服务器分别独立、并行地根据负载均衡策略进行数据转发和健康检查操作。如果某一台LVS节点服务器对后端某一台ECS健康检查失败，则该LVS节点服务器将不会再将新的客户端请求分发给相应的异常ECS。LVS集群内所有服务器同步进行该操作。

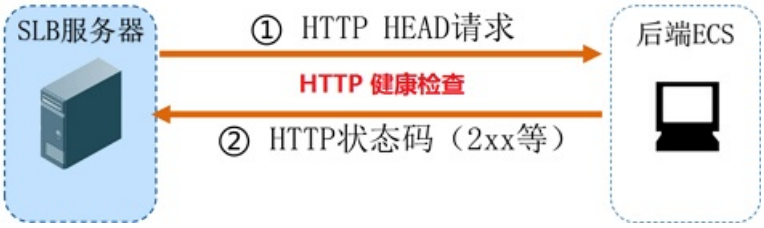
如下图所示，负载均衡健康检查使用的地址段是100.64.0.0/10，后端服务器务必不能屏蔽该地址段。您无需在ECS安全组中额外针对该地址段配置放行策略，但如有配置iptables等安全策略，请务必放行（100.64.0.0/10 是阿里云保留地址，其他用户无法分配到该网段内，不会存在安全风险）。




## HTTP/HTTPS监听健康检查机制

针对七层（HTTP或HTTPS协议）监听，健康检查通过HTTP HEAD探测来获取状态信息，如下图所示。

对于HTTPS监听，证书在负载均衡系统中进行管理。负载均衡与后端ECS之间的数据交互（包括健康检查数据和业务交互数据），不再通过HTTPS进行传输，以提高系统性能。



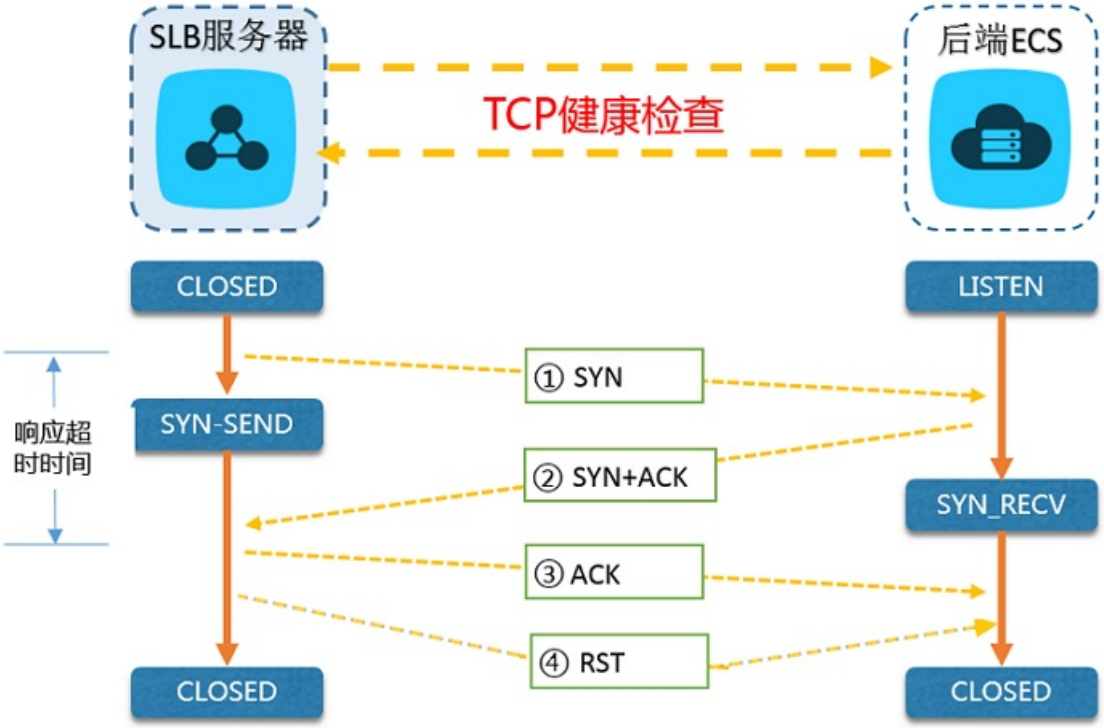

七层监听的检查机制如下：

1. Tengine节点服务器根据监听的健康检查配置，向后端ECS的内网IP+【健康检查端口】+【检查路径】发送HTTP HEAD请求（包含设置的【域名】）。

2. 后端ECS收到请求后，根据相应服务的运行情况，返回HTTP状态码。
3. 如果在【响应超时时间】之内，Tengine节点服务器没有收到后端ECS返回的信息，则认为服务无响应，判定健康检查失败。
4. 如果在【响应超时时间】之内，Tengine节点服务器成功接收到后端ECS返回的信息，则将该返回信息与配置的状态码进行比对。如果匹配则判定健康检查成功，反之则判定健康检查失败。

## TCP监听健康检查机制

针对四层TCP监听，为了提高健康检查效率，健康检查通过定制的TCP探测来获取状态信息，如下图所示。




TCP监听的检查机制如下：

1. LVS节点服务器根据监听的健康检查配置，向后端ECS的内网IP+【健康检查端口】发送TCP SYN数据包。
2. 后端ECS收到请求后，如果相应端口正在正常监听，则会返回SYN+ACK数据包。
3. 如果在【响应超时时间】之内，LVS节点服务器没有收到后端ECS返回的数据包，则认为服务无响应，判定健康检查失败；并向后端ECS发送RST数据包中断TCP连接。
4. 如果在【响应超时时间】之内，LVS节点服务器成功收到后端ECS返回的数据包，则认为服务正常运行，判定健康检查成功，而后向后端ECS发送RST数据包中断TCP连接。

**说明** 正常的TCP三次握手，LVS节点服务器在收到后端ECS返回的SYN+ACK数据包后，会进一步发送ACK数据包，随后立即发送RST数据包中断TCP连接。

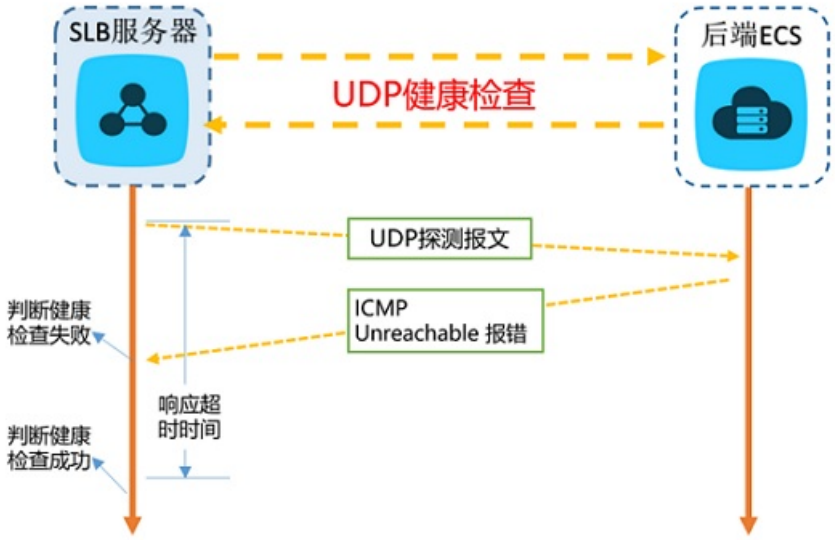
该实现机制可能会导致后端ECS认为相关TCP连接出现异常（非正常退出），并在业务软件如Java连接池等日志中抛出相应的错误信息，如 `Connection reset by peer` 。

解决方案：

- TCP监听采用HTTP方式进行健康检查。
- 在后端ECS配置了获取客户端真实IP后，忽略来自前述负载均衡服务地址段相关访问导致的连接错误。

## UDP监听健康检查

针对四层UDP监听，健康检查通过UDP报文探测来获取状态信息，如下图所示。




UDP监听的检查机制如下：

1. LVS节点服务器根据监听的健康检查配置，向后端ECS的内网IP+【健康检查端口】发送UDP报文。
2. 如果后端ECS相应端口未正常监听，则系统会返回类似 `port XX unreachable` 的ICMP报错信息，反之不做任何处理。
3. 如果在【响应超时时间】之内，LVS节点服务器收到了后端ECS返回的上述错误信息，则认为服务异常，判定健康检查失败。
4. 如果在【响应超时时间】之内，LVS节点服务器没有收到后端ECS返回的任何信息，则认为服务正常，判定健康检查成功。

> **说明** 当前UDP协议服务健康检查可能存在服务真实状态与健康检查不一致的问题：
>
> 如果后端ECS是Linux服务器，在大并发场景下，由于Linux的防ICMP攻击保护机制，会限制服务器发送ICMP的速度。此时，即便服务已经出现异常，但由于无法向前端返回 `port XX unreachable` 报错信息，会导致负载均衡由于没收到ICMP应答进而判定健康检查成功，最终导致服务真实状态与健康检查不一致。
>
> 解决方案：
>
> 负载均衡通过发送您指定的字符串到后端服务器，必须得到指定应答后才认为检查成功。但该实现机制需要客户端程序配合应答。

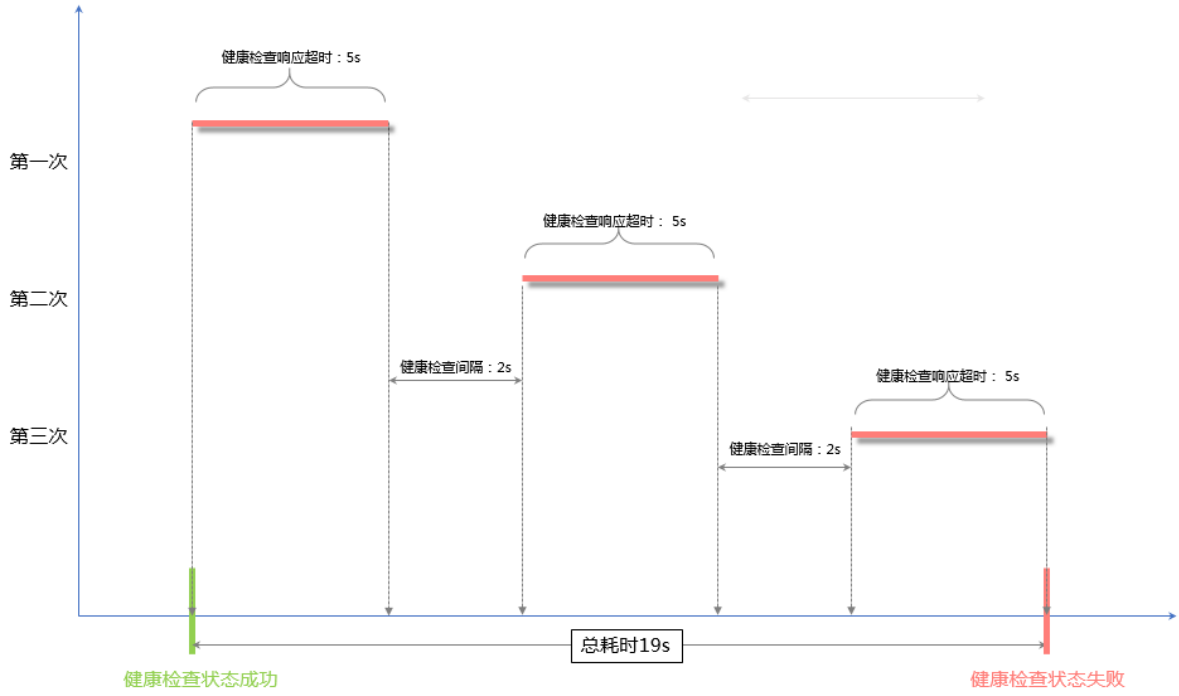
## 健康检查时间窗

健康检查机制的引入，有效提高了业务服务的可用性。但是，为了避免频繁的健康检查失败引起的切换对系统可用性的冲击，健康检查只有在健康检查时间窗内连续多次检查成功或失败后，才会进行状态切换。健康检查时间窗由以下三个因素决定：

- 健康检查间隔（每隔多久进行一次健康检查）
- 响应超时时间 （等待服务器返回健康检查的时间）
- 检查阈值（健康检查连续成功或失败的次数）

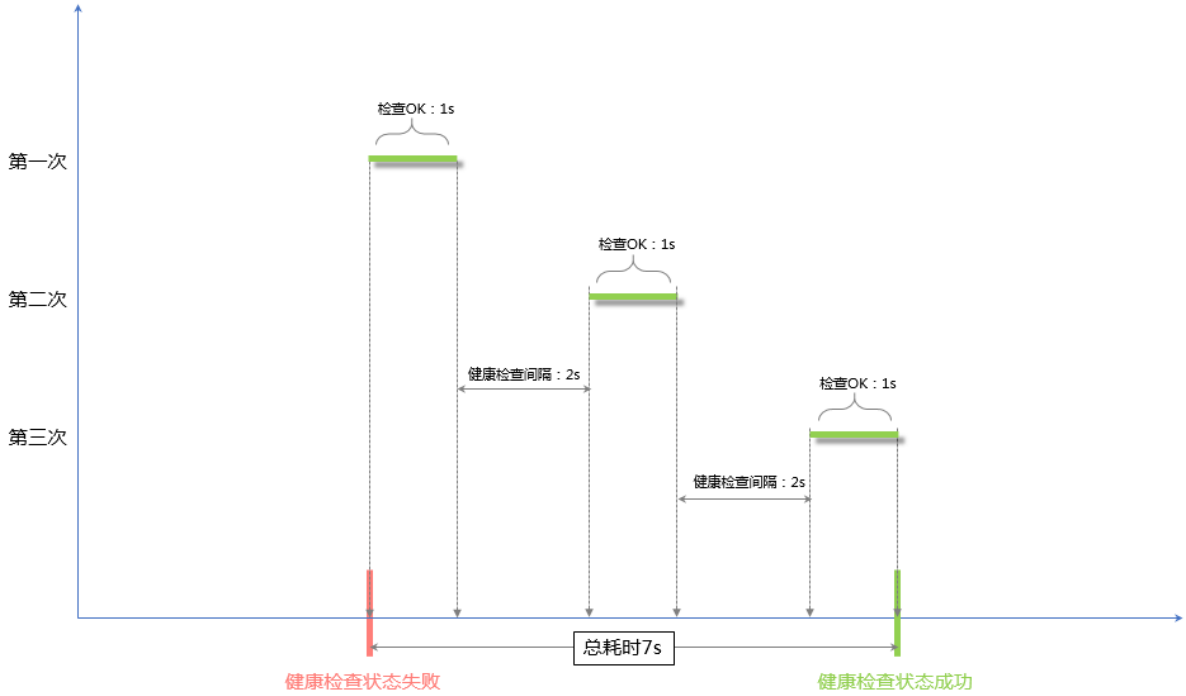
健康检查时间窗口的计算方法如下：

- 健康检查失败时间窗口=响应超时时间×不健康阈值+检查间隔×（不健康阈值-1）



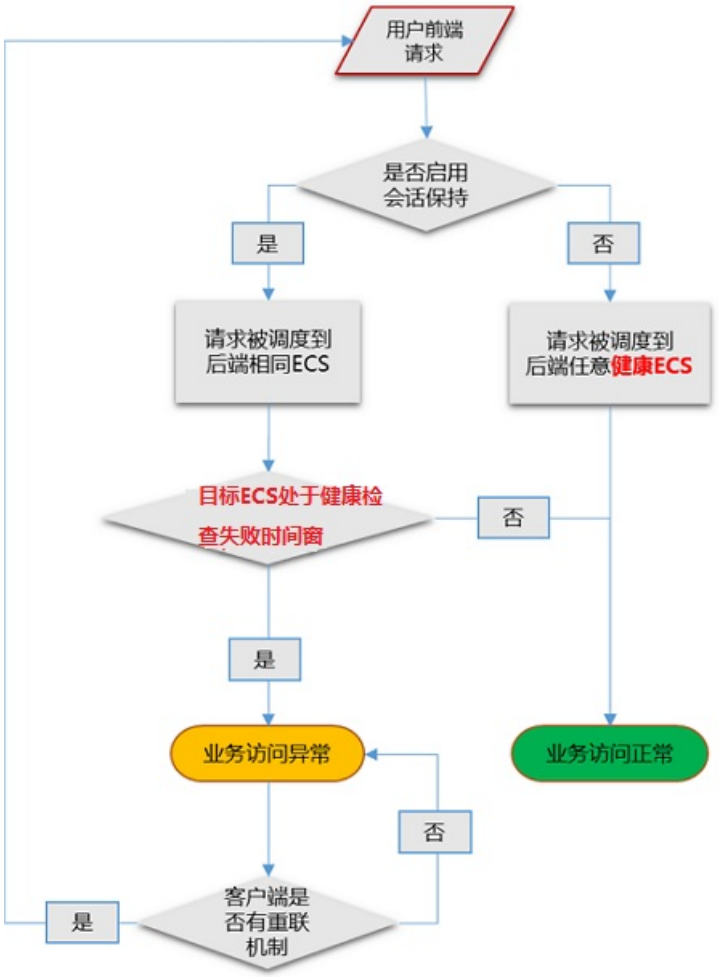

- 健康检查成功时间窗口= （健康检查成功响应时间x健康阈值）+检查间隔x（健康阈值-1）

> **说明** 健康检查成功响应时间是一次健康检查请求从发出到响应的时间。当采用TCP方式健康检查时，由于仅探测端口是否存活，因此该时间非常短，几乎可以忽略不计。当采用HTTP方式健康检查时，该时间取决于应用服务器的性能和负载，但通常都在秒级以内。




健康检查状态对请求转发的影响如下：

- 如果目标ECS的健康检查失败，新的请求不会再分发到相应ECS上，所以对前端访问没有影响。
- 如果目标ECS的健康检查成功，新的请求会分发到该ECS上，前端访问正常。
- 如果目标ECS存在异常，正处于健康检查失败时间窗，而健康检查还未达到检查失败判定次数（默认为三次），则相应请求还是会被分发到该ECS，进而导致前端访问请求失败。



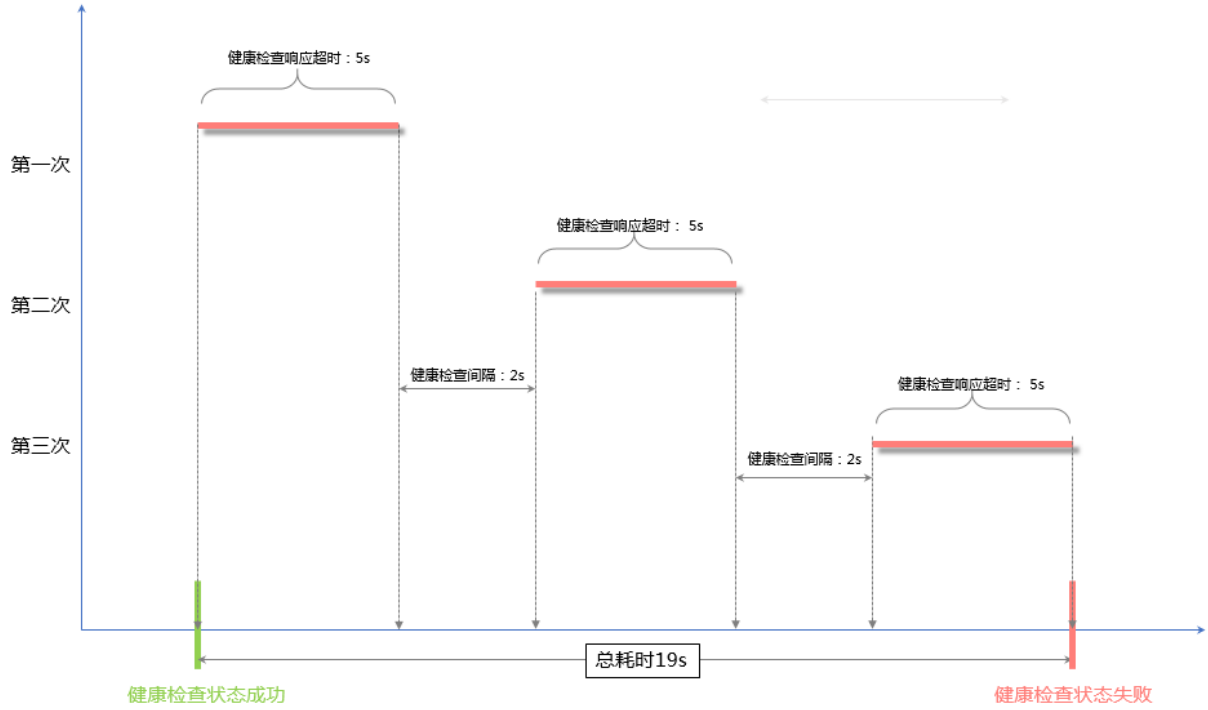

## 健康检查响应超时和健康检查间隔示例

以如下健康检查配置为例：

- **响应超时时间：5秒**
- **健康检查间隔：2秒**
- **健康阈值：3次**
- **不健康阈值：3次**

健康检查失败时间窗口=响应超时时间×不健康阈值+检查间隔×（不健康阈值-1），5×3+2×（3-1）=19s，即以19s为一个时间窗，健康检查响应时间超过19s，健康检查状态为不健康。

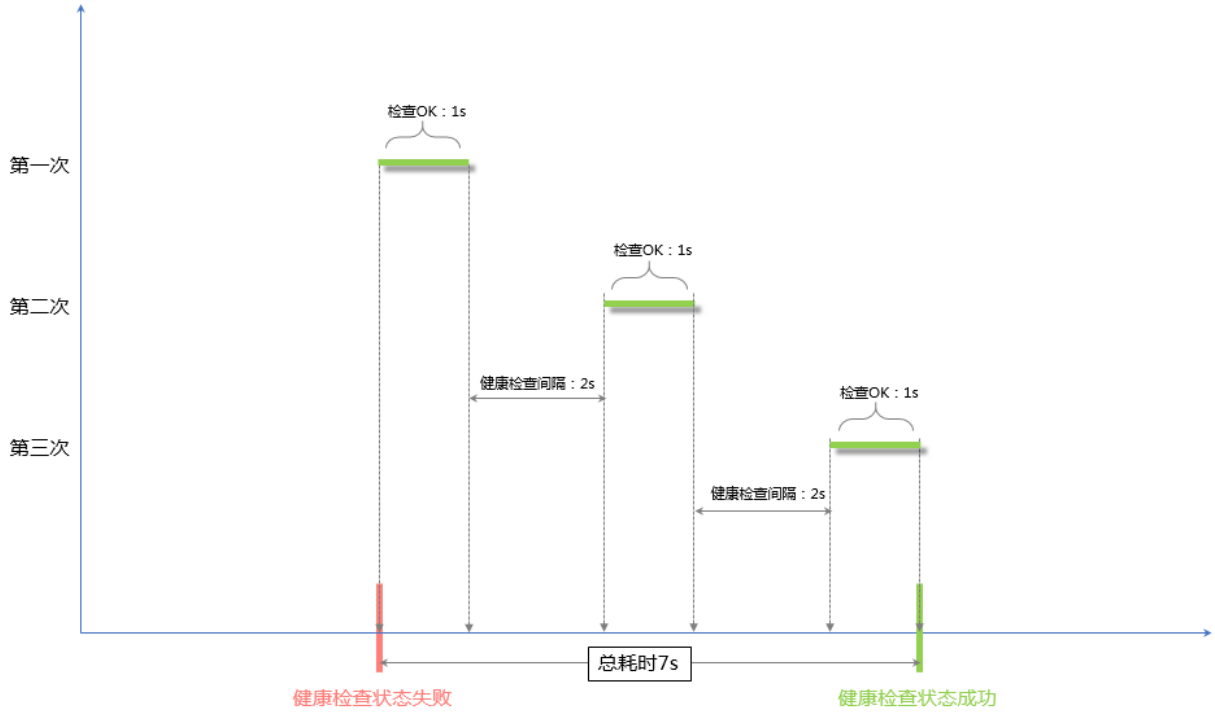
从健康状态到不健康状态的检查过程如下图所示：




健康检查成功时间窗口=（健康检查成功响应时间×健康阈值）+检查间隔×（健康阈值-1），（1×3）+2×（3-1）=7s，即以7s为一个时间窗，健康检查成功响应时间低于7s，健康检查状态为健康。

> **说明** 健康检查成功响应时间是一次健康检查请求从发出到响应的时间。当采用TCP方式健康检查时，由于仅探测端口是否存活，因此该时间非常短，几乎可以忽略不计。当采用HTTP方式健康检查时，该时间取决于应用服务器的性能和负载，但通常都在秒级以内。

从不健康状态到健康的状态检查过程如下图所示（假设服务器响应健康检查请求需要耗时1s）：



## HTTP健康检查中域名的设置

当使用HTTP方式进行健康检查时，可以设置健康检查的域名，但并非强制选项。因为有些应用服务器会对请求中的host字段做校验，即要求请求头中必须存在host字段。如果在健康检查中配置了域名，则SLB会将域名配置到host字段中去，反之，如果没有配置域名，SLB则不会在请求中附带host字段，因此健康检查请求就会被服务器拒绝，可能导致健康检查失败。综上原因，如果您的应用服务器需要校验请求的host字段，那么则需要配置相关的域名，确保健康检查正常工作。

# 2.配置健康检查

您可以在添加监听时配置健康检查，通常，使用默认的健康检查配置即可。

## 背景信息

您可以通过控制台或API配置监听的健康检查。更多详细信息，参见健康检查概述和健康检查常见问题。

## 操作步骤

1. 登录传统型负载均衡CLB控制台。
2. 选择地域，查看该地域的所有负载均衡实例。
3. 单击负载均衡实例的ID。
4. 单击**监听**页签。
5. 单击**添加监听**或目标监听**操作**列的**修改监听配置**。
6. 单击**下一步 至健康检查**页签，配置健康检查。

   在配置健康检查时，建议您使用默认值。

   健康检查配置说明

| 健康检查配置 | 说明 |
| --- | --- |
| **健康检查协议** | 选择健康检查协议类型，监听为TCP协议时，健康检查方式可选TCP或HTTP模式。<br>○ TCP模式的健康检查是基于网络层探测，通过发送SYN握手报文来检测服务器端口是否存活。<br>○ HTTP模式的健康检查是通过发送head请求，通过发送HEAD或GET请求模拟浏览器的访问行为来检查服务器应用是否健康。 |
| **健康检查方法**<br>（仅HTTP和HTTPS健康检查协议支持） | 七层监听（HTTP/HTTPS）健康检查支持HEAD和GET方法，默认采用HEAD方法。<br>如果您的后端应用服务器不支持HEAD方法或HEAD方法被禁用，则可能会出现健康检查失败，此时可以使用GET方法来进行健康检查。<br>使用GET方法时，如果Response长度超过8K，会被截断，但不会影响健康检查结果的判定。<br>**说明** 所有地域的七层监听健康检查都支持GET方法。 |
| **健康检查路径和健康检查域名（可选）**<br>（仅HTTP健康检查协议支持） | HTTP健康检查默认由负载均衡系统通过后端ECS内网IP地址向该服务器应用配置的缺省首页发起HTTP Head请求。<br>如果您用来进行健康检查的页面并不是应用服务器的缺省首页，需要指定具体的检查路径。<br>因为有些应用服务器会对请求中的host字段做校验，即要求请求头中必须存在host字段。如果在健康检查中配置了域名，则SLB会将域名配置到host字段中去，反之，如果没有配置域名，SLB则不会在请求中附带host字段，因此健康检查请求就会被服务器拒绝，可能导致健康检查失败。综上原因，如果您的应用服务器需要校验请求的host字段，则需要配置相关域名，确保健康检查正常工作。 |

| 健康检查配置 | 说明 |
| --- | --- |
| **正常状态码**<br>（仅HTTP健康检查协议支持） | 选择健康检查正常的HTTP状态码。<br>默认值为http_2xx和http_3xx。 |
| **健康检查端口** | 健康检查服务访问后端时的探测端口。<br>默认值为配置监听时指定的后端端口。<br>**说明** 如果该监听配置了虚拟服务器组或主备服务器组，且组内的ECS实例的端口都不相同，此时不需要配置检查端口。负载均衡系统会使用各自ECS的后端端口进行健康检查。 |
| **健康检查响应超时时间** | 接收来自运行状况检查的响应需要等待的时间。如果后端ECS在指定的时间内没有正确响应，则判定为健康检查失败。<br>范围是1~300秒，UDP监听的默认值为10秒，HTTP/HTTPS/TCP监听的默认值为5秒。 |
| **健康检查间隔时间** | 进行健康检查的时间间隔。<br>LVS集群内所有节点，都会独立、并行地遵循该属性对后端ECS进行健康检查。由于各LVS节点的检查时间并不同步，所以，如果从后端某一ECS上进行单独统计，会发现来自负载均衡的健康检查请求在时间上并不会遵循上述时间间隔。<br>范围是1-50秒，UDP监听的默认值为5秒，HTTP/HTTPS/TCP监听的默认值为2秒。 |
| **健康检查不健康阈值** | 同一LVS节点服务器针对同一ECS服务器，从成功到失败的连续健康检查失败次数。<br>可选值2-10，默认为3次。 |
| **健康检查健康阈值** | 同一LVS节点服务器针对同一ECS服务器，从失败到成功的连续健康检查成功次数。<br>可选值 2-10，默认为3次。 |
| **健康检查请求**和**健康检查返回结果** | 为UDP监听配置健康检查时，您可以在**健康检查请求**中输入请求的内容（例如**youraccountID**），在**健康检查返回结果**中输入预期的返回结果（例如**slb123**）。<br>同时在后端服务器的应用逻辑中加入相应的健康检查应答逻辑，如收到youraccountID的请求时，回应slb123。<br>此时，当负载均衡收到后端服务器发来的正确响应时，则认为健康检查成功，否则认为健康检查失败。此方式能最大程度确保健康检查的可靠性。 |

7. 单击**下一步**。

## 相关文档

- 健康检查探测

# 3.健康检查探测

健康检查探测是根据负载均衡监听中的健康检查配置生成探测脚本，通过ECS云助手在您的ECS实例上执行脚本，获取健康探测结果，用于在您配置后端服务器后提前探测后端服务器的健康状态。

## 前提条件

您需要满足以下条件，才能对后端服务器进行健康检查探测：

- 确保您已经授权负载均衡对ECS实例进行健康检查探测，才能使用健康检查探测功能。授权请单击RAM角色授权。
- 后端服务器必须是VPC类型并且安装了Linux系统和云助手的ECS实例，实例处于running状态，系统默认Shell为bash。
- 负载均衡监听已开启健康检查并且后端服务器组中已经添加ECS实例。

## 背景信息

健康检查探测相关说明如下：

- 目前不支持对转发规则的后端服务器进行探测。
- 由于健康检查探测和实际健康检查上报采用了不同的链路，所以最终的结果可能不完全一致，探测为用户提供健康检查配置上的建议，而健康检查以配置完成后实际结果为准。

## 操作步骤

1. 登录传统型负载均衡CLB控制台。
2. 在**实例管理**页面，单击目标实例ID。
3. 在**监听**页签下，找到目标监听，在**操作**列单击**修改监听配置**。
4. 在**配置监听**页面，单击**下一步 至健康检查**。
5. 单击**高级配置**后的**健康检查探测**。
6. 在**健康检查探测**页面，单击目标后端服务器操作列下的**开始探测**。

   负载均衡支持同时选择5个ECS实例进行批量探测。如果ECS数量过大，请分批进行探测。


7. 单击**确定**开始诊断。诊断完成后控制台将展示诊断结果。

   目前监听支持的诊断项如下：

| 监听类型 | 健康检查端口状态 | iptables配置 | rpfilter配置 | HTTP探测响应 | UDP探测 |
| --- | --- | --- | --- | --- | --- |
| TCP | √ | √ | √ | √ | - |
| UDP | √ | √ | √ | - | √ |
| HTTP | √ | √ | √ | √ | - |
| HTTPS | √ | √ | √ | √ | - |

您可以登录云助手，选择ECS实例所在的地域，单击**执行记录**页签，查询探测脚本执行详情。

# 4.关闭健康检查

您可以关闭健康检查功能，但关闭健康检查后，当后端某个ECS出现异常时，

CLB

还是会把请求转发到该异常的ECS上，造成部分业务不可访问。所以建议一般情况下不要关闭健康检查。

## 操作步骤

1. 登录传统型负载均衡CLB控制台。
2. 在**实例管理**页面，单击目标

   CLB

   实例ID。
3. 在**监听**页签下，在目标监听的**操作**列单击**修改监听配置**。
4. 在**配置监听**页面，单击**下一步**至**健康检查**。
5. 关闭健康检查开关，单击**下一步**。
6. 单击**提交**，然后单击**知道了**。

# 5.健康检查FAQ

文本介绍传统型负载均衡CLB（Classic Load Balancer）健康检查相关的常见问题。



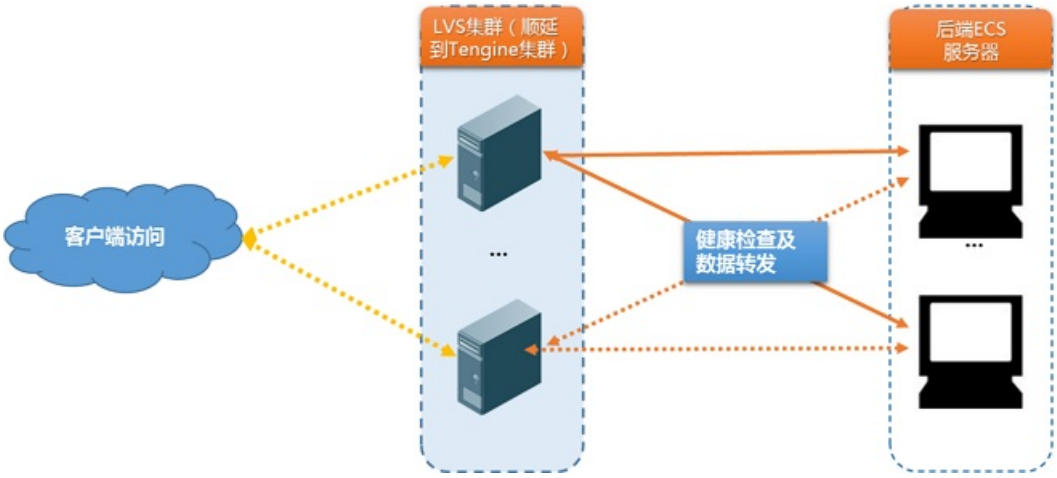
## 健康检查的原理是什么？

负载均衡通过健康检查来探测后端ECS实例的可用性。开启健康检查功能后，当后端某个ECS实例健康检查出现异常时，来自客户端的新请求将不会再被转发到该ECS实例，直到该ECS实例恢复正常。

负载均衡健康检查使用的地址段是100.64.0.0/10，后端服务器不能屏蔽该地址段。您无需在ECS安全组中额外针对该地址段配置放行策略，但如有配置iptables等安全策略，请保证已放行100.64.0.0/10地址段。

> **说明** 100.64.0.0/10是阿里云保留地址段，其他用户不会分配到该网段内，放行该地址段不会存在安全风险。

更多信息，请参见健康检查概述。




## 推荐的健康检查配置是什么？

为了避免由于健康检查频繁失败引起的切换对系统可用性造成的冲击，健康检查只有在健康检查时间窗内连续多次检查成功或失败后，才会进行状态切换。更多信息，请参见配置健康检查。

以下是TCP、HTTP和HTTPS监听建议使用的健康检查配置。

| 配置 | 推荐值 |
| --- | --- |
| **响应超时时间** | 5秒 |
| **健康检查间隔** | 2秒 |
| **不健康阈值** | 3 |

以下是UDP监听建议使用的健康检查配置。

| 配置 | 推荐值 |
| --- | --- |
| **响应超时时间** | 10秒 |
| **健康检查间隔** | 5秒 |
| **不健康阈值** | 3 |
| **健康阈值** | 3 |

**说明** 此配置有利于您的服务和应用状态的尽快收敛。如果您有更高要求，可以适当地降低响应超时时间值，但必须优先保证服务在正常状态下的处理时间小于这个值。

## 是否可以关闭健康检查？

可以关闭。具体操作，请参见关闭健康检查。

**说明** 如果关闭健康检查，当后端某个ECS实例健康检查出现异常时，负载均衡还是会把请求转发到该异常的ECS实例上，造成部分业务不可访问。不建议您关闭健康检查。

## TCP监听如何选择健康检查方式？

TCP监听支持HTTP和TCP两种健康检查方式：

- TCP协议健康检查通过发送SYN握手报文，检测服务器端口是否存活。
- HTTP协议健康检查通过发送HEAD或GET请求，模拟浏览器的访问行为来检查服务器应用是否健康。

TCP健康检查方式对服务器的性能资源消耗相对要少一些。如果您对后端服务器的负载高度敏感，则选择TCP健康检查；如果负载不是很高，则选择HTTP健康检查。

## ECS实例权重设置为零对健康检查有什么影响？

该状态下，负载均衡不会再将流量转发给该ECS实例，监听的后端服务器健康检查不会显示异常。

将负载均衡后端ECS实例的权重置为零，相当于将该ECS实例移出负载均衡。一般是在ECS实例进行重启和配置调整等主动运维时将其权重设置为零。

## HTTP监听向后端ECS实例执行健康检查使用的方法是什么？

HEAD方法。

如果后端ECS实例的服务关闭HEAD方法，会导致健康检查失败。建议在ECS实例上用HEAD方法访问自己IP地址进行测试：

```
curl -v -0 -I -H "Host:" -X HEAD http://IP:port
```

## HTTP监听向后端ECS实例执行健康检查的IP地址是什么？

负载均衡健康检查使用的地址段是100.64.0.0/10，后端服务器不能屏蔽该地址段。您无需在ECS安全组中额外针对该地址段配置放行策略，但如有配置iptables等安全策略，请保证已放行100.64.0.0/10地址段。

> **说明** 100.64.0.0/10是阿里云保留地址段，其他用户不会分配到该网段内，放行该地址段不会存在安全风险。

## 为什么健康检查监控频率与Web日志记录不一致？

负载均衡健康检查服务也是集群方式的，这样可以避免单点故障。负载均衡的代理分布到很多节点上，因此看到的健康检查日志访问频率和控制台设置的频率不一致，这是正常现象。

## 负载均衡因后端数据库故障导致健康检查失败，如何处理？

- **问题现象**

  ECS实例内配置了两个网站：`www.example.com`是静态网站，`aliyundoc.com`是动态网站，都配置了负载均衡。后端数据库服务异常，导致访问`www.example.com`静态网站出现502错误。

- **问题原因**

  负载均衡健康检查配置的检查域名是`aliyundoc.com`，RDS或者自建数据库故障导致`aliyundoc.com`访问异常，所以健康检查失败。

- **解决方案**

  将负载均衡健康检查域名配置为`www.example.com`即可。

## 负载均衡服务TCP端口健康检查成功，为什么在后端业务日志中出现网络连接异常信息？

- **问题现象**

  负载均衡后端配置TCP服务端口后，后端业务日志中频繁出现类似如下网络连接异常错误信息。经过抓包分析，发现相关请求来自负载均衡服务器，同时负载均衡主动向服务器发送了RST数据包。

```
java.io.IOException: Connection reset by peer
        at sun.nio.ch.FileDispatcherImpl.read0(Native Method)
        at sun.nio.ch.SocketDispatcher.read(SocketDispatcher.java:39)
        at sun.nio.ch.IOUtil.readIntoNativeBuffer(IOUtil.java:223)
        at sun.nio.ch.IOUtil.read(IOUtil.java:192)
        at sun.nio.ch.SocketChannelImpl.read(SocketChannelImpl.java:379)
        at io.netty.buffer.UnpooledUnsafeDirectByteBuf.setBytes(UnpooledUnsafeDirectByteBuf.java:446)
        at io.netty.buffer.AbstractByteBuf.writeBytes(AbstractByteBuf.java:871)
        at io.netty.channel.socket.nio.NioSocketChannel.doReadBytes(NioSocketChannel.java:225)
        at io.netty.channel.nio.AbstractNioByteChannel$NioByteUnsafe.read(AbstractNioByteChannel.java:115)
        at io.netty.channel.nio.NioEventLoop.processSelectedKey(NioEventLoop.java:507)
        at io.netty.channel.nio.NioEventLoop.processSelectedKeysOptimized(NioEventLoop.java:464)
        at io.netty.channel.nio.NioEventLoop.processSelectedKeys(NioEventLoop.java:378)
        at io.netty.channel.nio.NioEventLoop.run(NioEventLoop.java:350)
        at io.netty.util.concurrent.SingleThreadEventExecutor$2.run(SingleThreadEventExecutor.java:116)
        at java.lang.Thread.run(Thread.java:745)
```

- **问题原因**

  该问题和负载均衡的健康检查机制有关。

  由于TCP对上层业务状态无感知，同时，为了降低负载均衡健康检查成本和对后端业务的冲击，当前负载均衡针对TCP协议服务端口的健康检查只会做简单的TCP三次握手，而后直接发送RST包断开TCP连接。数据交互流程如下：

i. 负载均衡服务器向后端负载均衡服务端口发送SYN请求包。
ii. 后端服务器收到请求后，如果端口状态正常，则按照正常的TCP机制返回相应的SYN+ACK应答包。
iii. 负载均衡服务器成功收到后端服务端口应答后，则认为端口监听是正常的，判定健康检查成功。
iv. 负载均衡服务器向相应TCP服务端口直接发送RST包主动关闭连接，结束本次健康检查操作，且没有继续发送业务数据。

如上所述，由于健康检查成功后，负载均衡服务器直接发送TCP RST包中断了连接，并没有做进一步的业务数据交互，导致上层业务（例如Java连接池等）认为相应的连接是异常的，所以会出现 `Connection reset by peer` 等错误信息。

- **解决方案**
  - 更换TCP协议为HTTP协议。
  - 在业务层面，对来自SLB服务器IP地址段的相关请求做日志过滤，忽略相关错误信息。

## 为什么业务本身没有异常但是健康检查显示异常？

- **问题现象**

  负载均衡HTTP方式的健康检查始终失败，但测试 `curl -I` 得到的状态码是正常的。

```
echo -e 'HEAD /test.html HTTP/1.0\r\n\r\n' | nc -t 192.168.0.1 80
```

- **问题原因**

  如果返回的状态与控制台配置的正常状态码不一致，则判定健康检查异常。如果您配置的正常状态码为http_2xx，则所有返回的非HTTP 2xx状态码均被认为是健康检查失败。

  Tengine/Nginx配置执行 `curl` 命令会发现没有问题，但是执行 `echo` 命令会匹配到默认站点，导致测试文件*test.html*返回404错误，如下图所示。

```
[root@iZ28s[redacted]Z home]# echo -e "HEAD /test.html HTTP/1.0\r\n\r\n" |nc -t 10.161.93.136 80
HTTP/1.1 404 Not Found
Server: Tengine/2.1.0
Date: Mon, 16 Feb 2015 07:29:32 GMT
Content-Type: text/html
Content-Length: 585
Connection: close

[root@iZ28[redacted]Z home]# curl -I http://10.161.93.136/test.html
HTTP/1.1 200 OK
Server: Tengine/2.1.0
Date: Mon, 16 Feb 2015 07:29:41 GMT
Content-Type: text/html
Content-Length: 5
Last-Modified: Mon, 16 Feb 2015 07:27:00 GMT
Connection: keep-alive
ETag: "54e19bc4-5"
Accept-Ranges: bytes
```

- **解决方案**
  - 修改主配置文件，注释默认站点。
  - 在健康检查配置中添加检查域名。