杭州微碟科技有限公司

EIOC04 用户手册 V1.0

2016-3-15

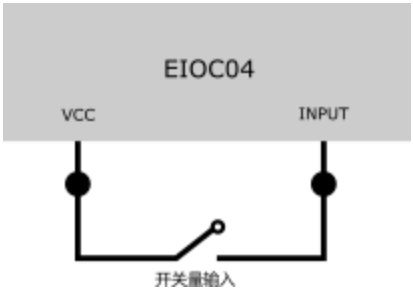
目录

简介

EIOC04 以太网 IO 控制器包括 4 路继电器输出，4 路开关量输入，支持 9~24V 宽电压供电。提供开关量继电器联动功能、远端点对点控制功能。提供一个 Restful 的 HTTP API 接口和一个 MODBUS TCP 接口。

接口说明

VCC、INPUT 构成一组开关量输入。注意，开关量输出只允许接入干节点。开关量输入请按下图所示，



RESET 为复位按钮，长按 10 秒以上，控制器配置参数会回到默认值。

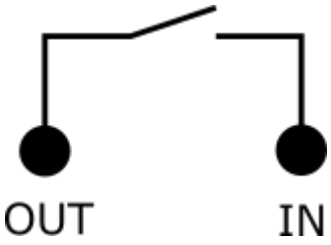
OUT 为继电器输出；

IN 为继电器输入；

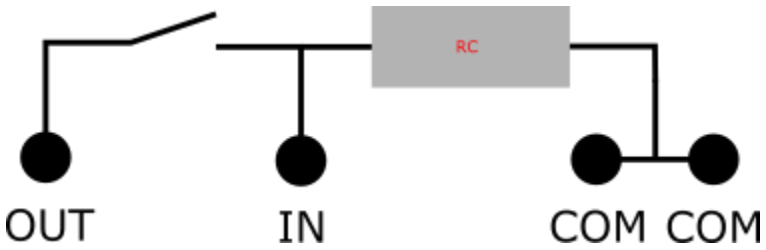
COM 为 RC 吸收模块的公共端；

A、B 为配线端子。

EIOC4 型号中 RELAY1、2 为标准的继电器输出，结构图如下，



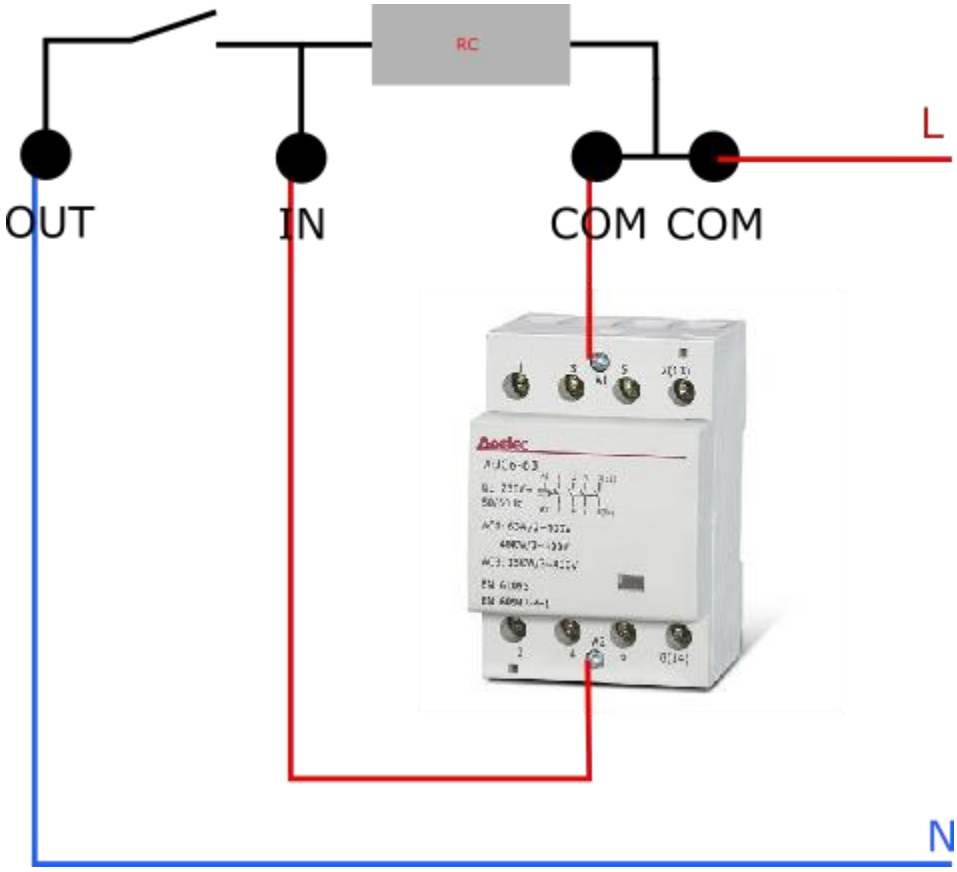
RELAY3、4 为带有 RC 吸收电路（120 欧，0.1uF）的继电器输出，结构图如下，



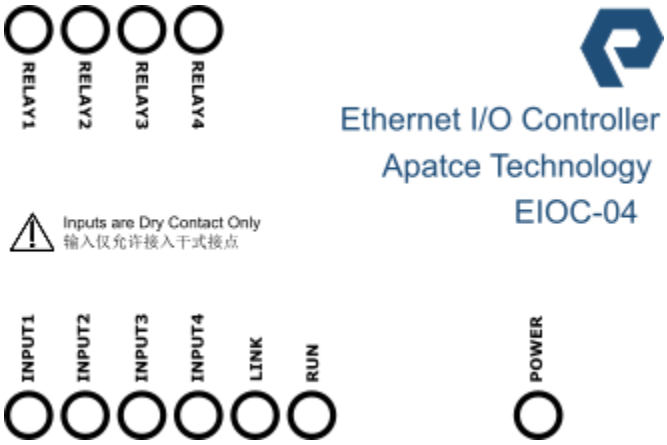
A、B 为配线端子，可提供方便的电线并接，结构图如下，

注意，对于 RELAY1、2 建议仅作为开关量输出，如果控制电源请添加相应的保护电路。

RELAY3、4 作为电源控制时，建议按如下方式接线：



LED 指示说明



RELAY 为继电器输出指示灯，当继电器闭合相应的指示灯会亮起。

INPUT 为开关量输入指示灯，当开关闭合相应的指示灯会亮起。

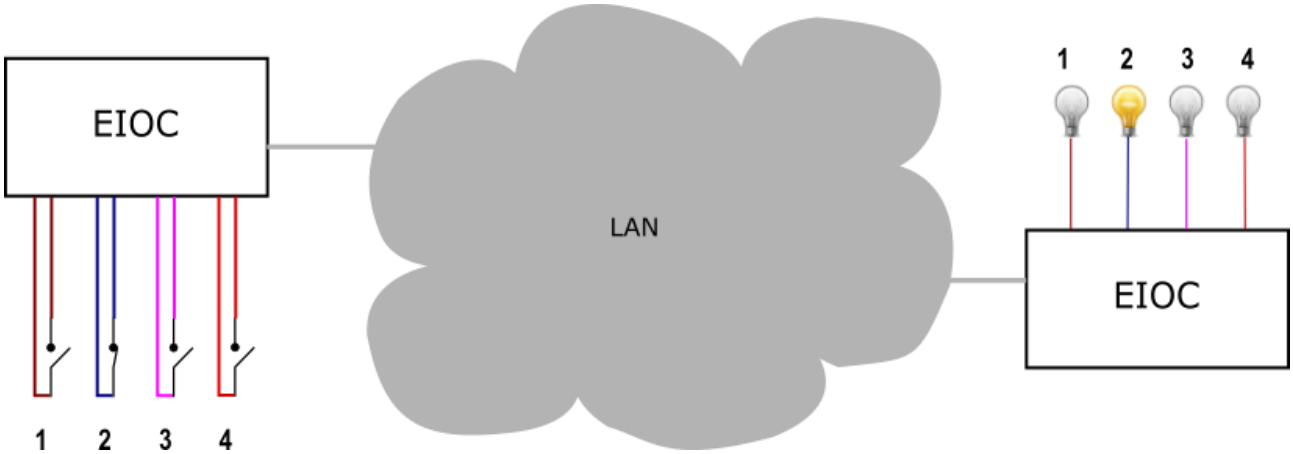
LINK 为远端控制指示灯，当控制器连接到远端控制器时，该指示灯为亮起，如果无法连接或连接断开后该指示灯为熄灭。

RUN 为工作指示灯，当控制器正常工作时，RUN 指示会按 1 秒的频繁闪烁。当按下 RESET 后，该指示灯会快闪，当按下 10 秒以上后，该指示灯会长亮，并完成复位。

POWER 为电源指示灯，插入电源后该指示灯会亮起。

远端控制

EIOC04 控制器支持远端点对点控制，即通过一个本地 EIOC 控制器的开关量输入控制远端的 EIOC 控制器，远端控制器的继电器输出状态将会和本地 EIOC 控制器的开关量状态一致。



如上图所示，当第 2 路开关量输入闭合后，远端的 EIOC 控制器的第 2 路输出将会打开。

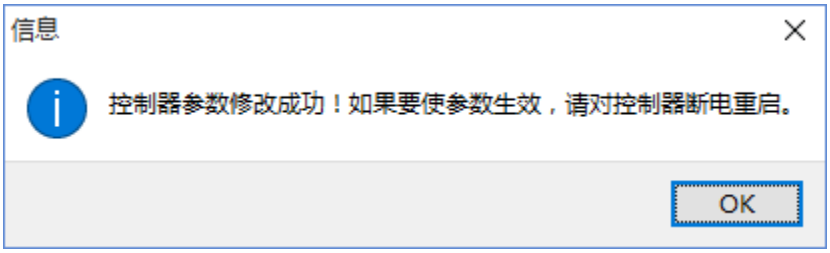
参数配置

控制器可以通过 EIOCSetting 软件进行配置，也可以通过 HTTP API 进行配置。使用 HTTP API 进行配置请参阅 HTTP API 一节。本节讲述如何使用 EIOCSetting 进行配置。

EIOCSetting 简要说明

启动 EIOCSetting 后在控制器 IP 处输入控制器 IP 地址，输入控制器访问密码后点击“连接”按钮。

软件会自动载入当前控制器的配置参数。
完成参数修改后点击“提交”按钮，保存成功后，软件出弹出提示框。



注意：任何参数的修改必须对控制器断电重启后才能生效。

参数说明

除 IP，掩码，网关和密码等基本参数外，控制器还包括 Modbus，远端 IP，联动，断电记忆等特别参数，本节具体讲述这些参数的作用。

Modbus

当钩选后 Modbus 功能会被起用，当取消勾选后 Modbus 功能会被禁止。
注意：禁止 Modbus 后将无法通过 EIOC 进行远端控制。

远端 IP

远端控制器的 IP 地机，控制器会在上电时连接该控制器，并会将自身的开关量状态传输给该 IP 的控制器实现远端控制。当远端 IP 地址配为 0.0.0.0 时表示关闭远端控制器。

断电记忆

当钩选后控制器会记录断电时的继电器状态，并在供电恢复时自动将继电器状态回到断电前。

联动功能

当钩选后，开关量输入的改变会造成继电器输出的反转。

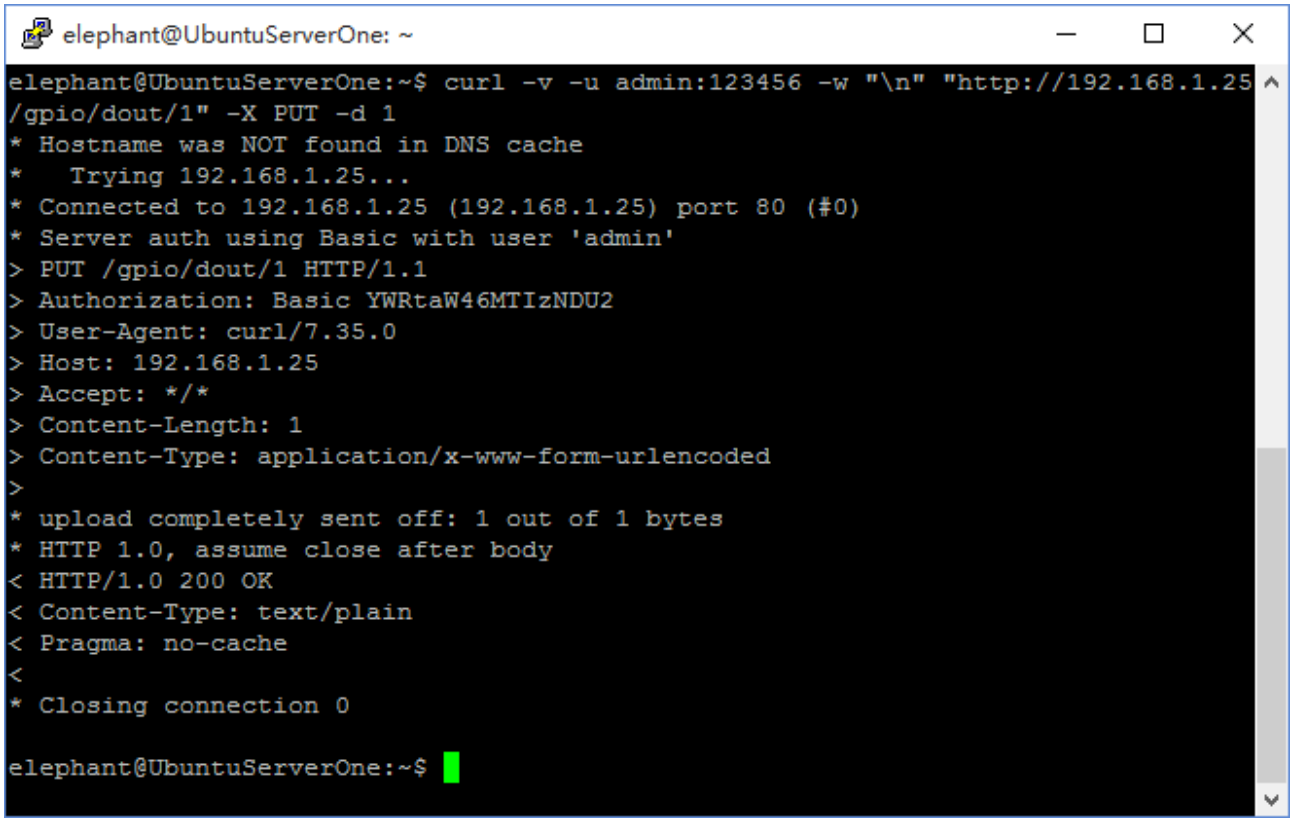
HTTP API

EIOC 系列的 HTTP API 是完全 Restful 形式，支持 HTTP Basic 认证，使用 GET 方法获取状态和参数，使用 PUT 或 POST 方法设置状态和参数。

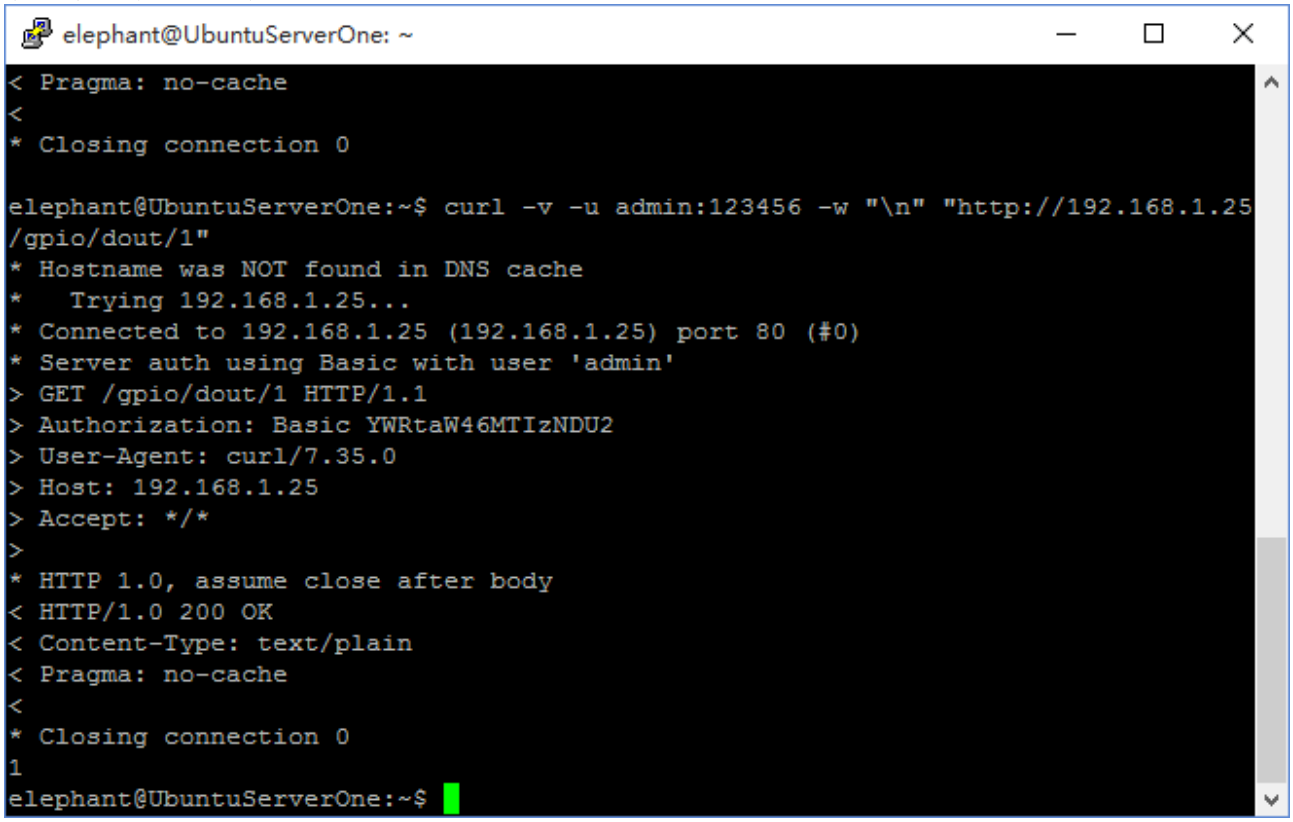
EIOC04 的 HTTP API 参考 draft-ipso-app-framework-04，对单一资源的操作返回数据格式为 text/plain，对批量资源的参考 draft-jennings-senml-10，访问返回 JSON 格式数据。
EIOC 的默认 IP 为 192.168.1.25，HTTP Basic 认证的用户名为（并且仅有）Admin，默认密码为 123456，本节将使用这些默认参数进行讲述。
在本节中我们使用 CURL 这一命令行工具进行演示。

继电器

资源路径：/gpio/dout/(1...4)，资源 GET 为获取状态，PUT 为控制输出
url 示例: http://192.168.1.25/gpio/dout/1
设置第 1 路继电器闭合



获取第 1 路继电器状态

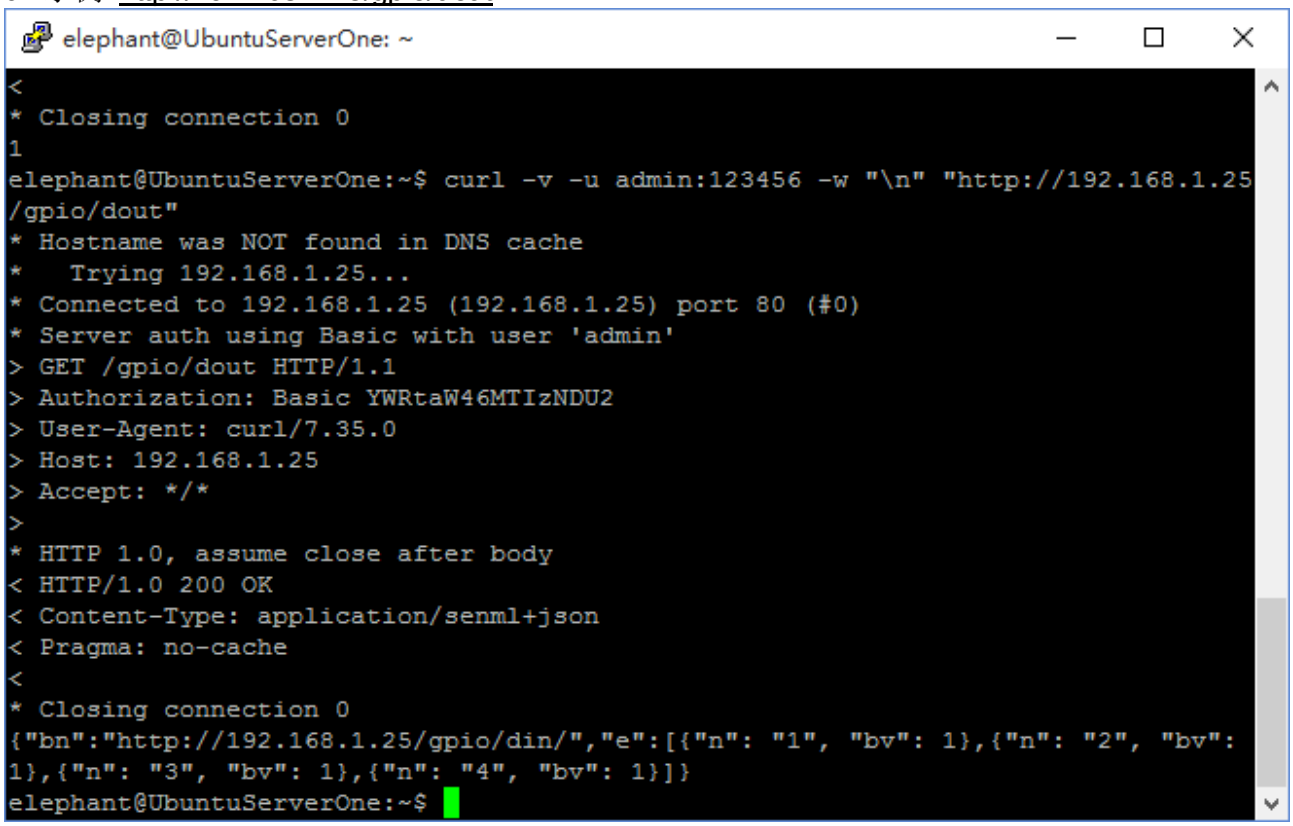


批量获取继电器状态

批量获取继电器状态只支持 GET 模式，返回的数据格式为 JSON 格式。

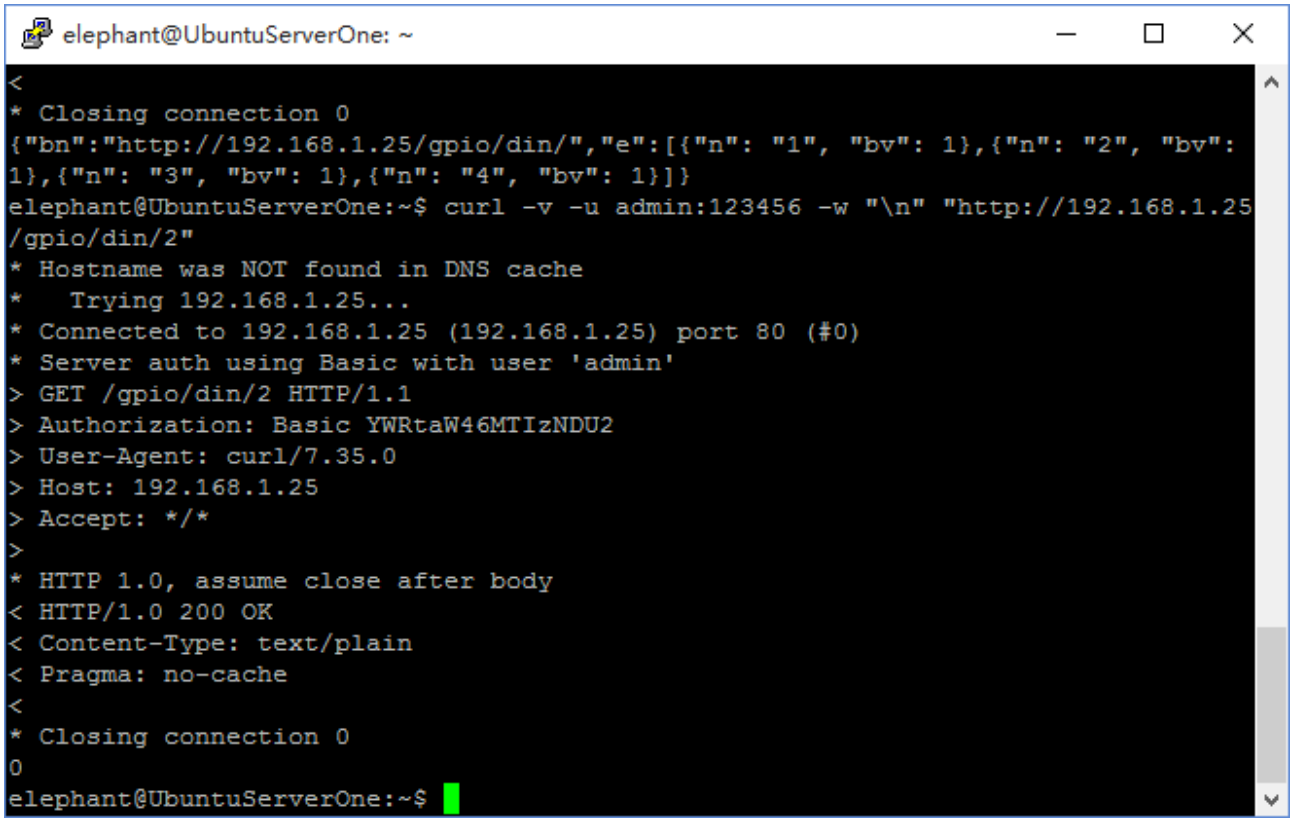
资源路径：/gpio/dout

url 示例: http://192.168.1.25/gpio/dout



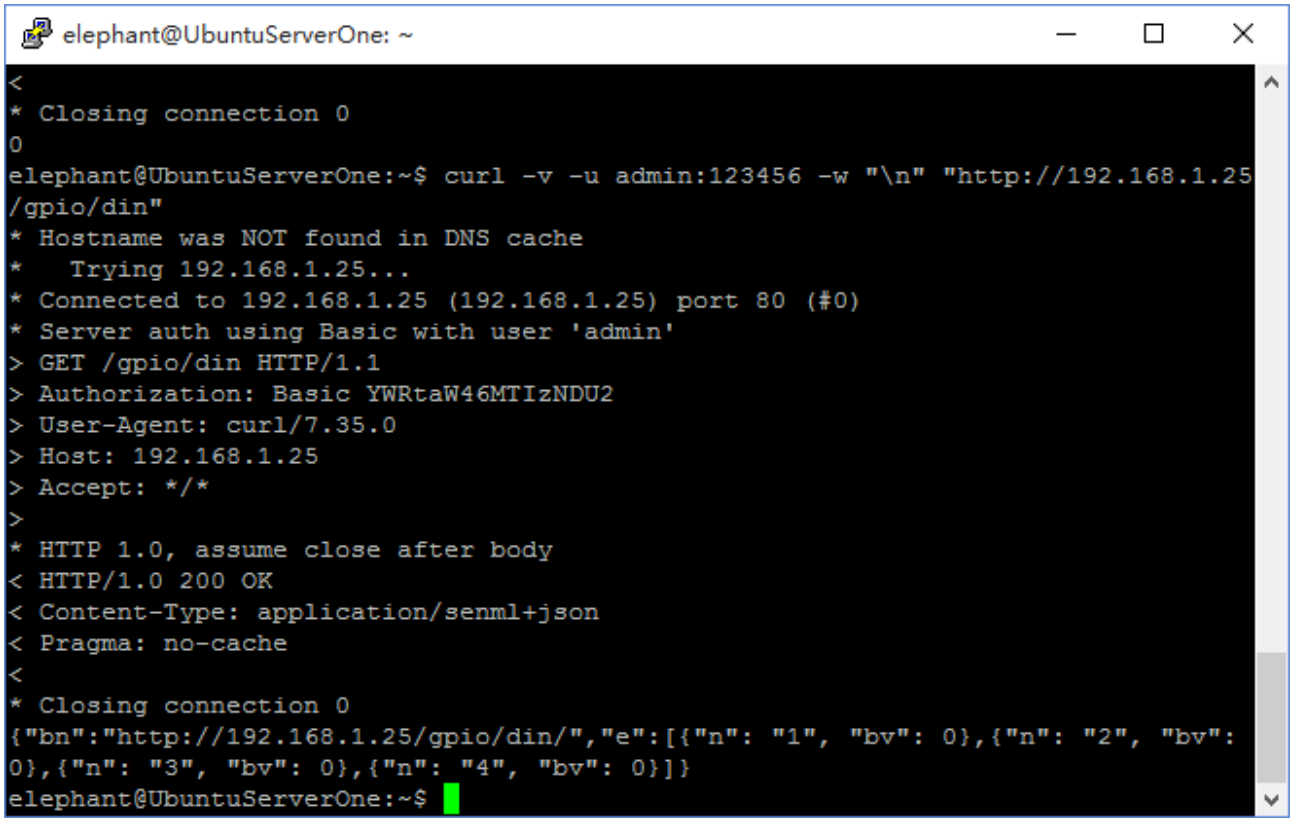
开关量

资源路径：/gpio/din/(1...4)，该资源仅支持 GET 方式
url 示例：http://192.168.1.25/gpio/din/2
获取第 2 路开关量的输入状态



批量获取开关量输入状态

批量获取开关量状态只支持 GET 模式，返回的数据格式为 JSON 格式。
资源路径：/gpio/din
url 示例: http://192.168.1.25/gpio/din

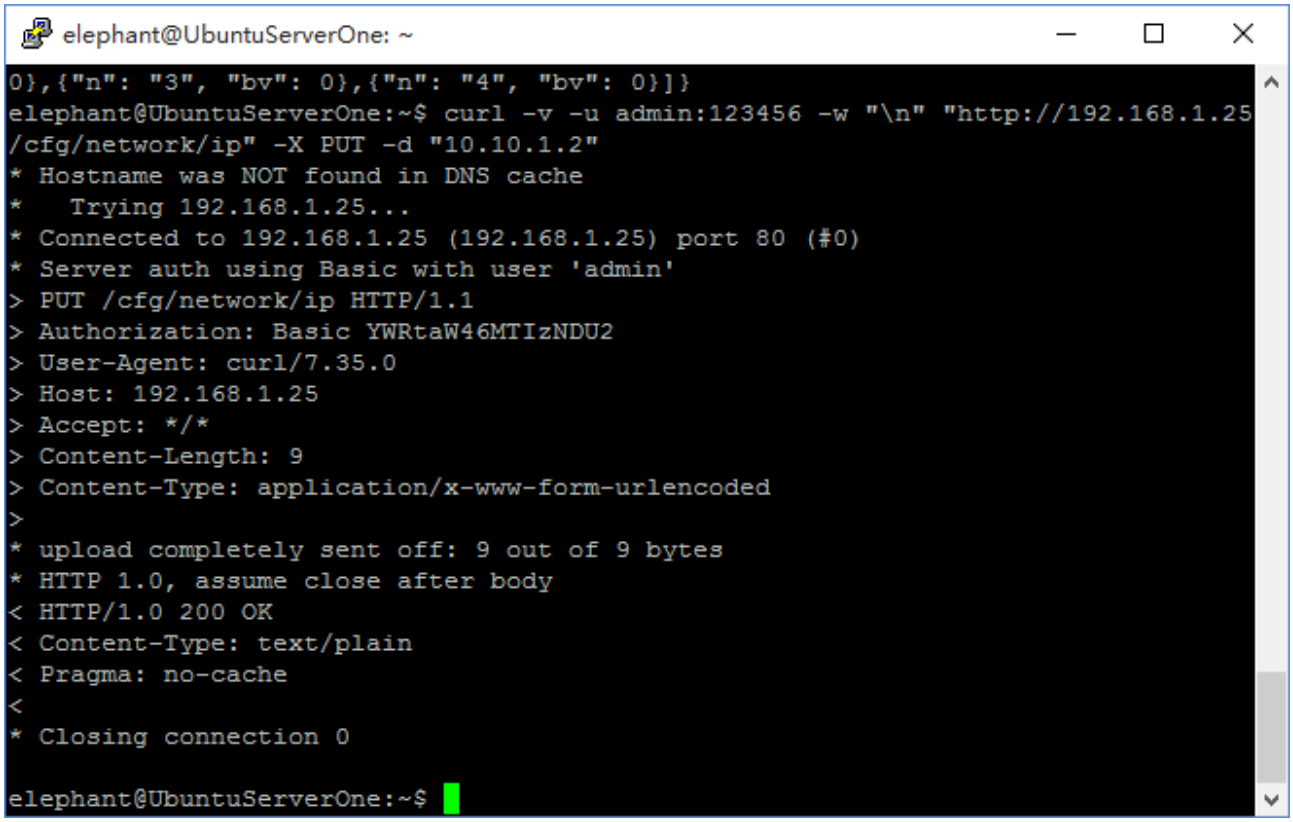


控制器参数

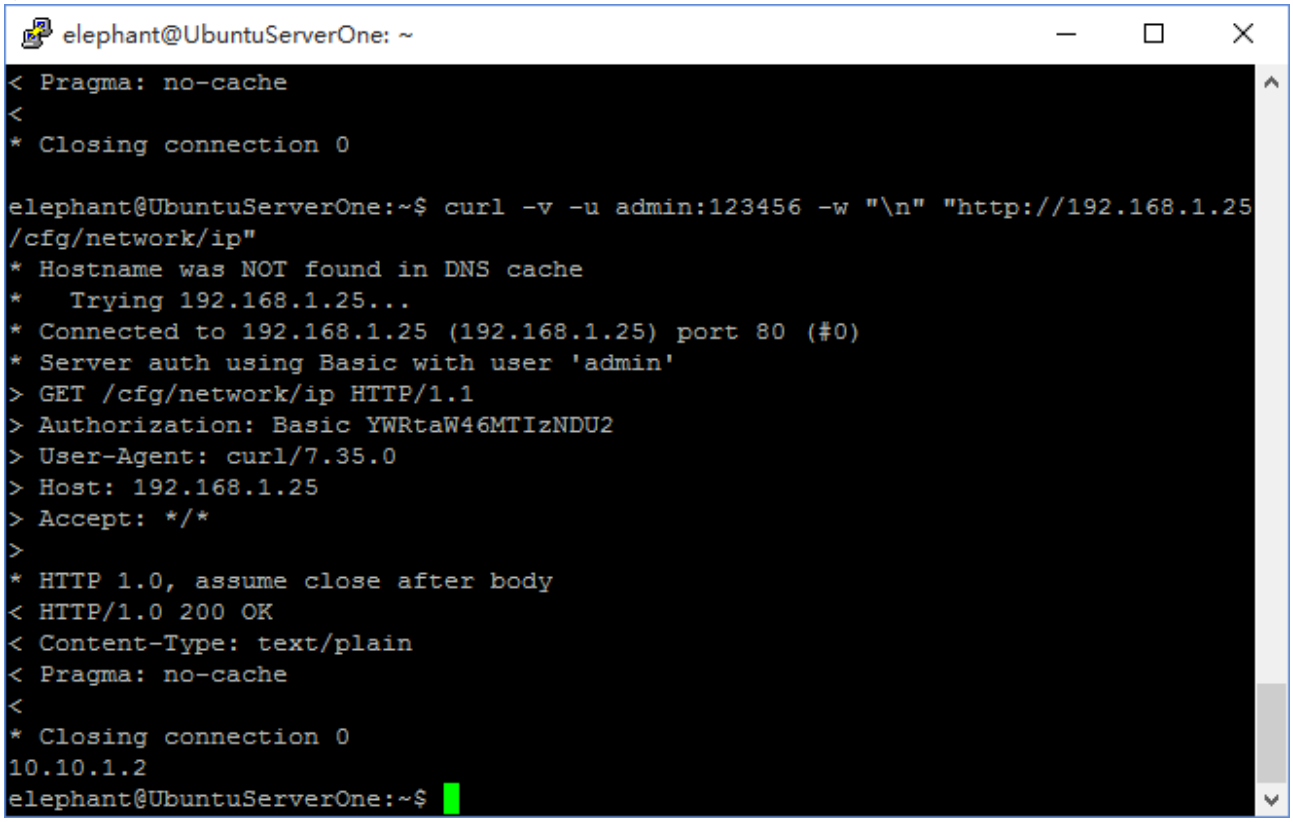
注意，所有对控制器参数的修改必须断电重启才能生效。

IP

资源路径：/cfg/network/ip
url 示例：http://192.168.1.25/cfg/network/ip
修改控制器 IP 为 10.10.1.2



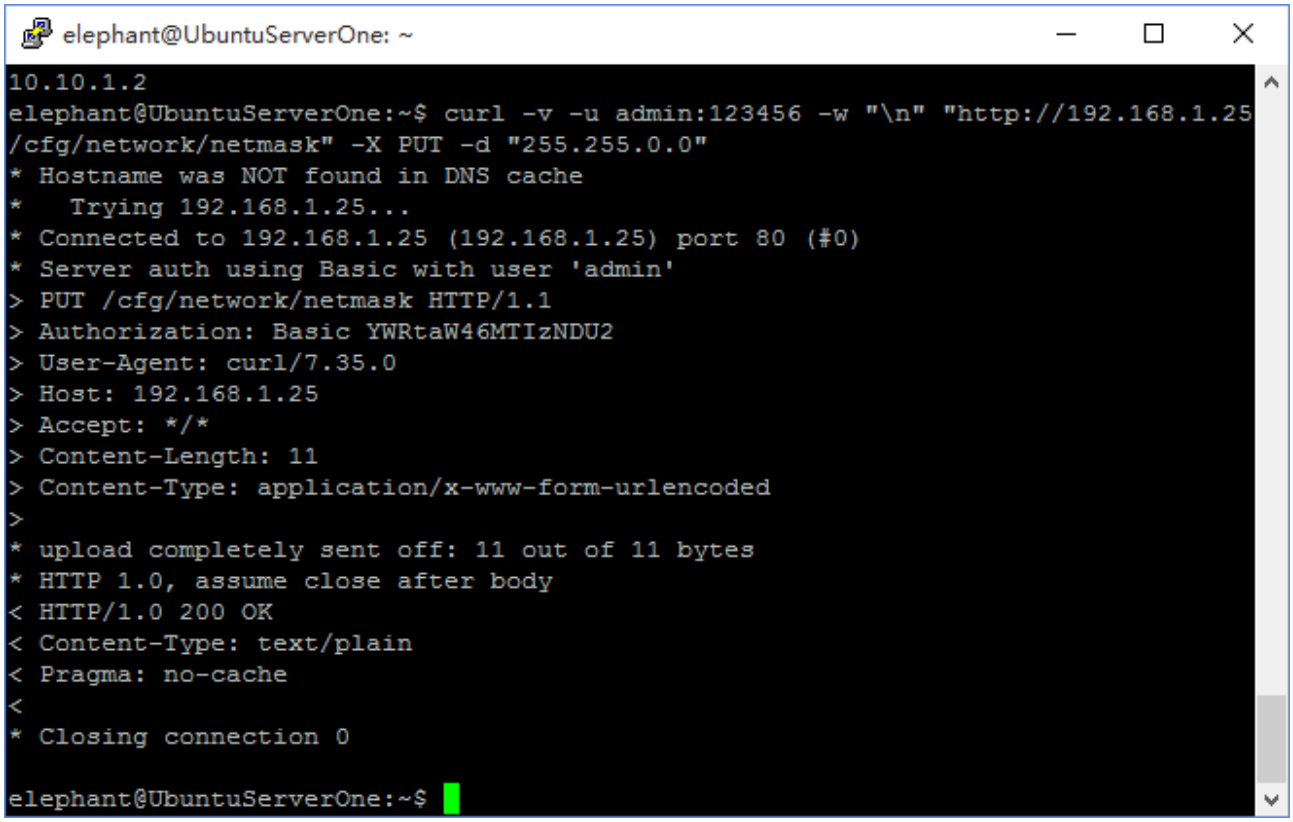
获取控制器配置的 IP 地址



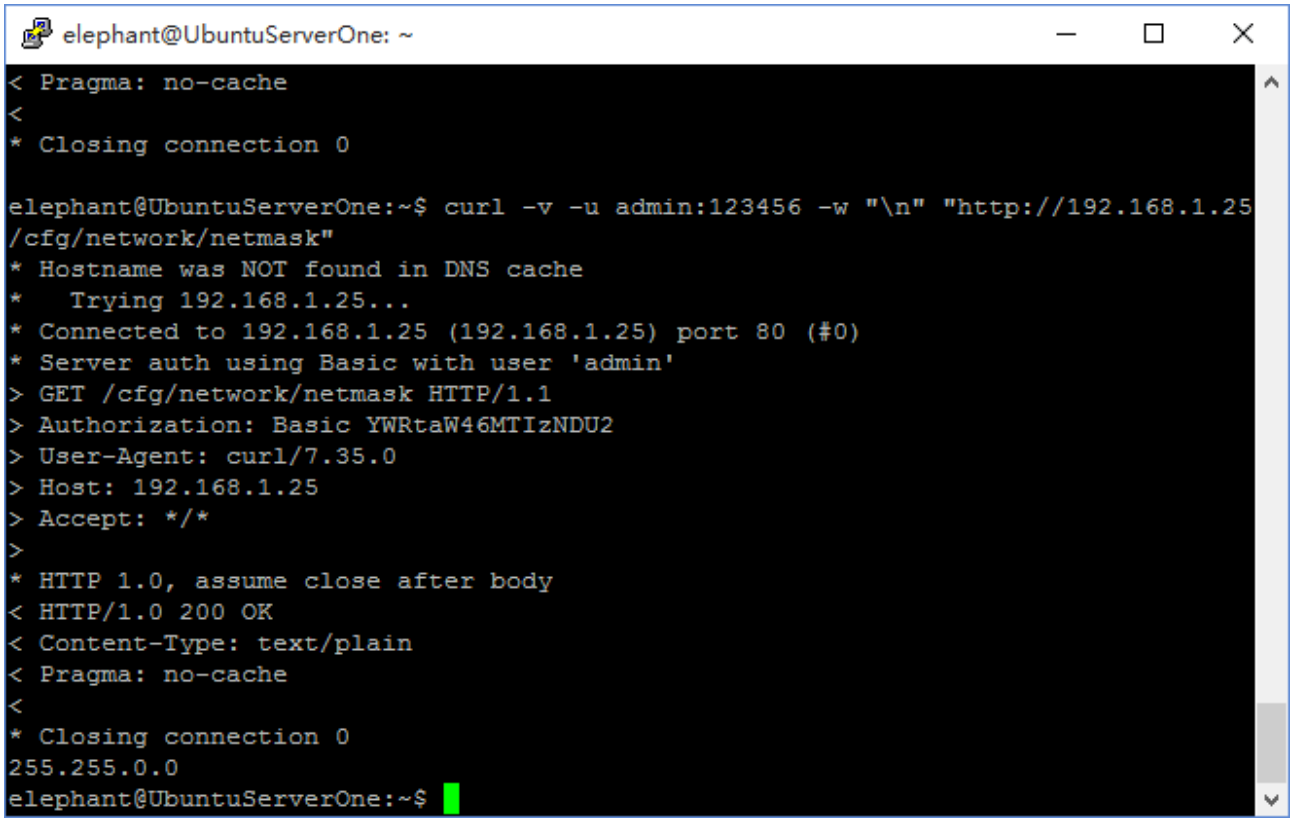
Netmask

资源路径：/cfg/network/netmask
url 示例：http://192.168.1.25/cfg/network/netmask

修改控制器的掩码地址为 255.255.0.0



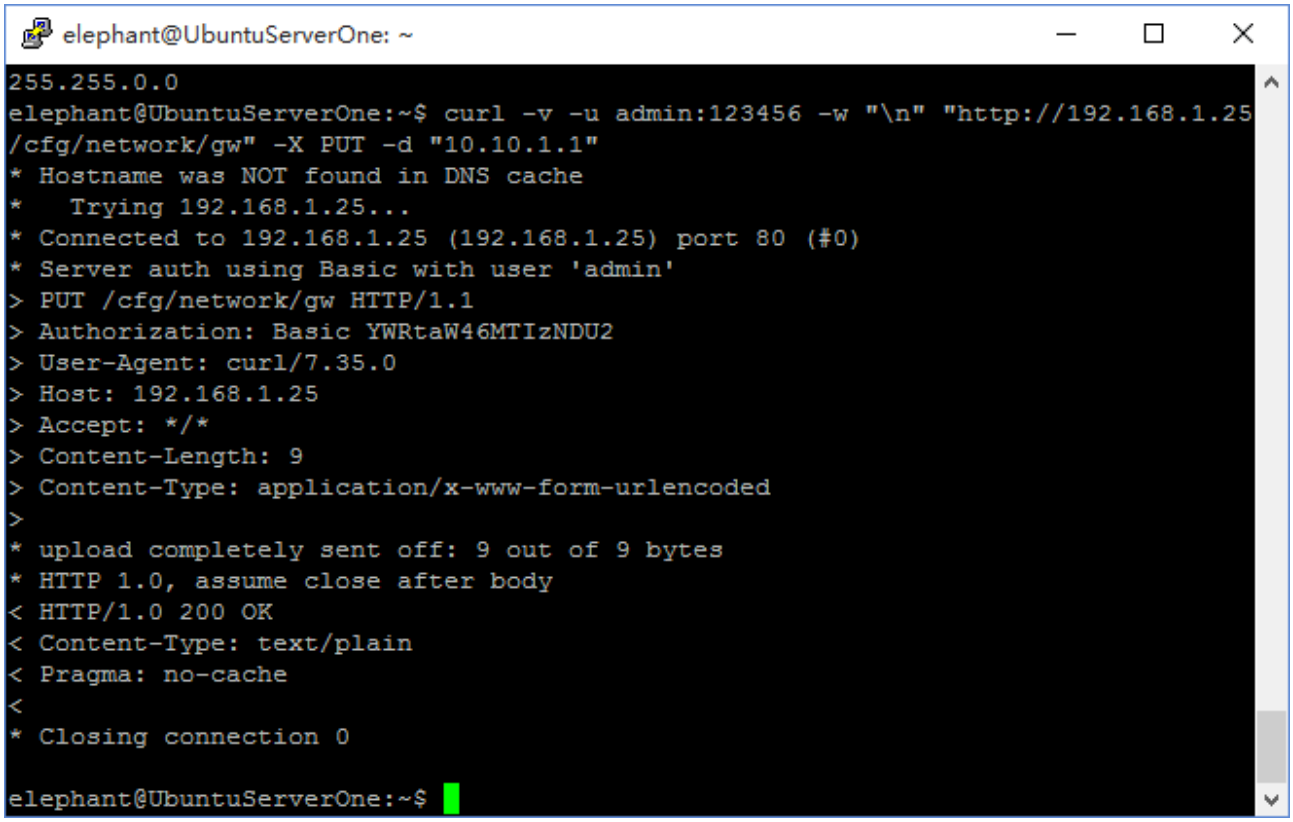
获取控制器配置的掩码地址



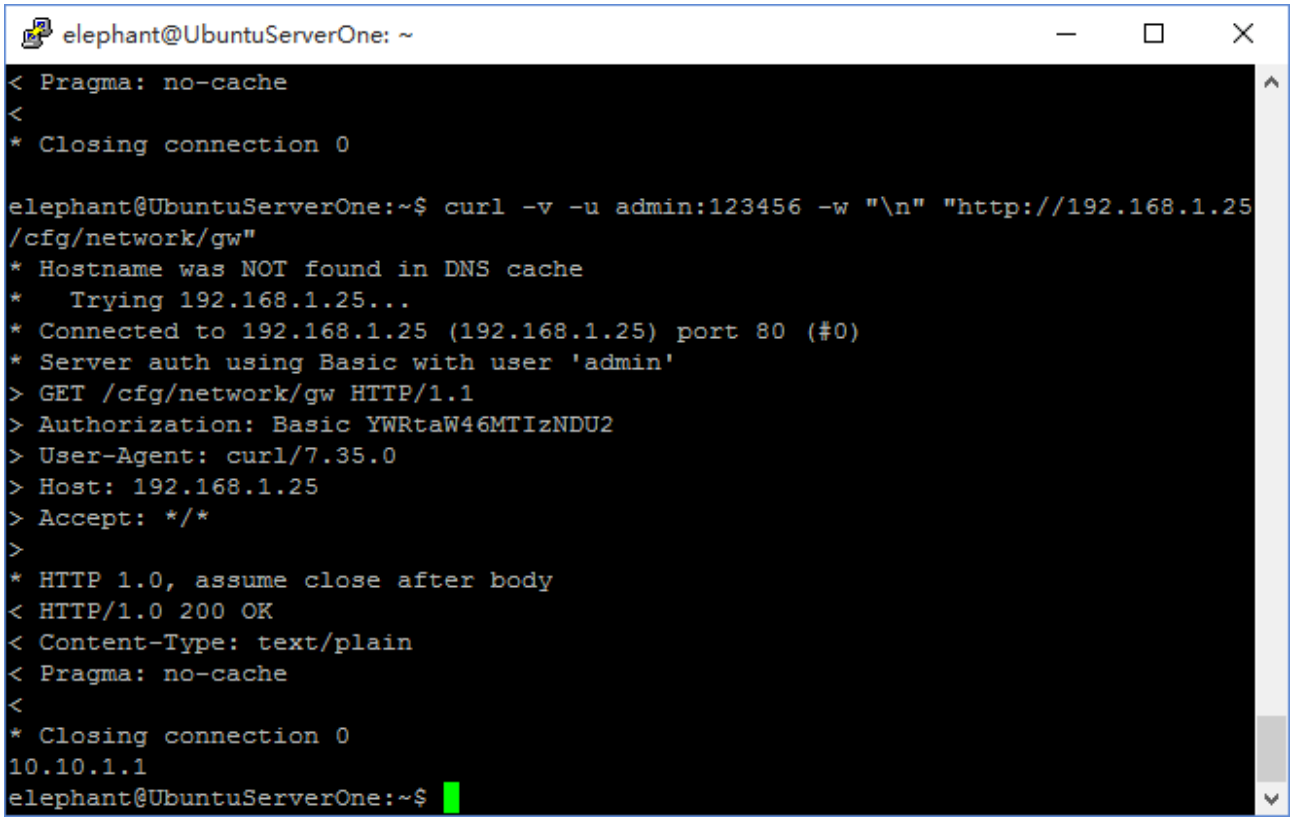
Gateway

资源路径：/cfg/network/gw

url 示例：http://192.168.1.25/cfg/network/gw
修改控制器的网关地址为 10.10.1.1



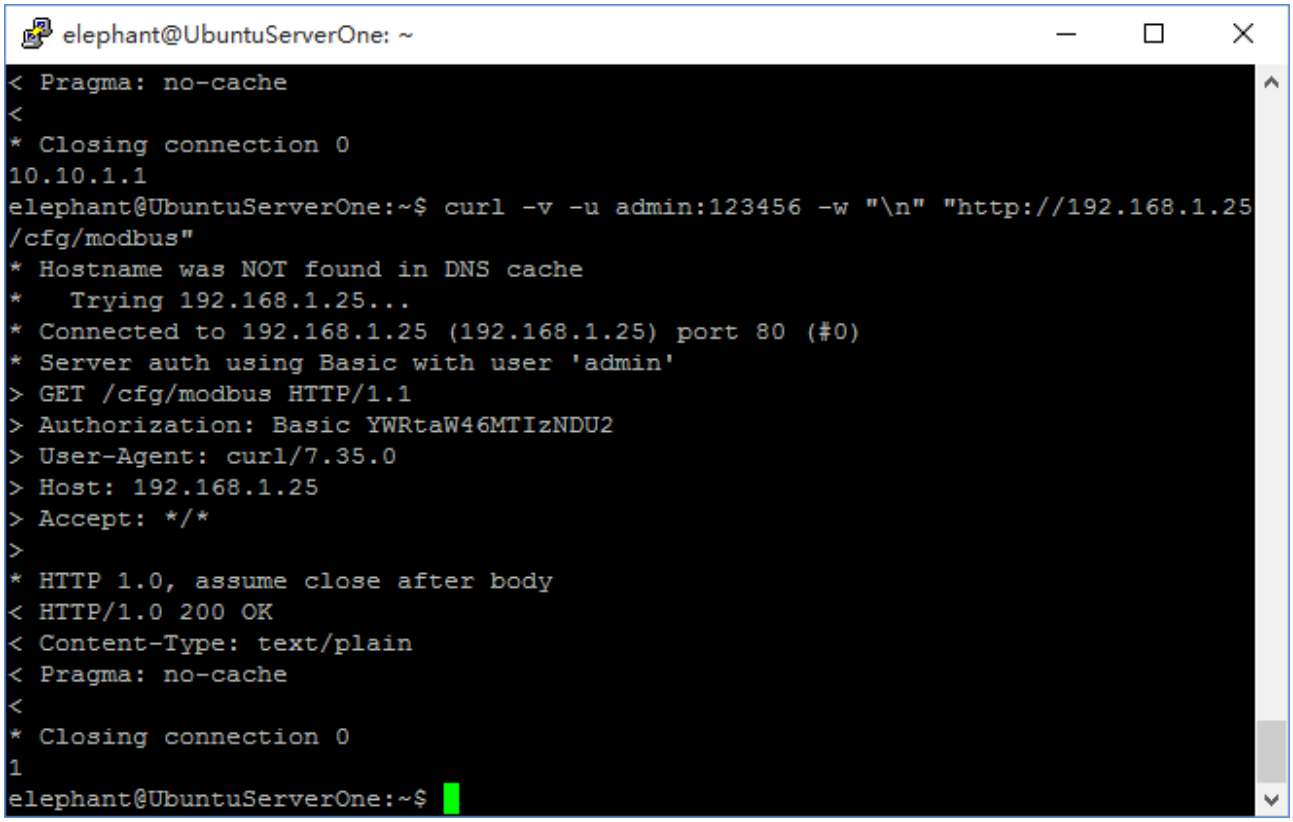
获取当前控制器配置的网关地址



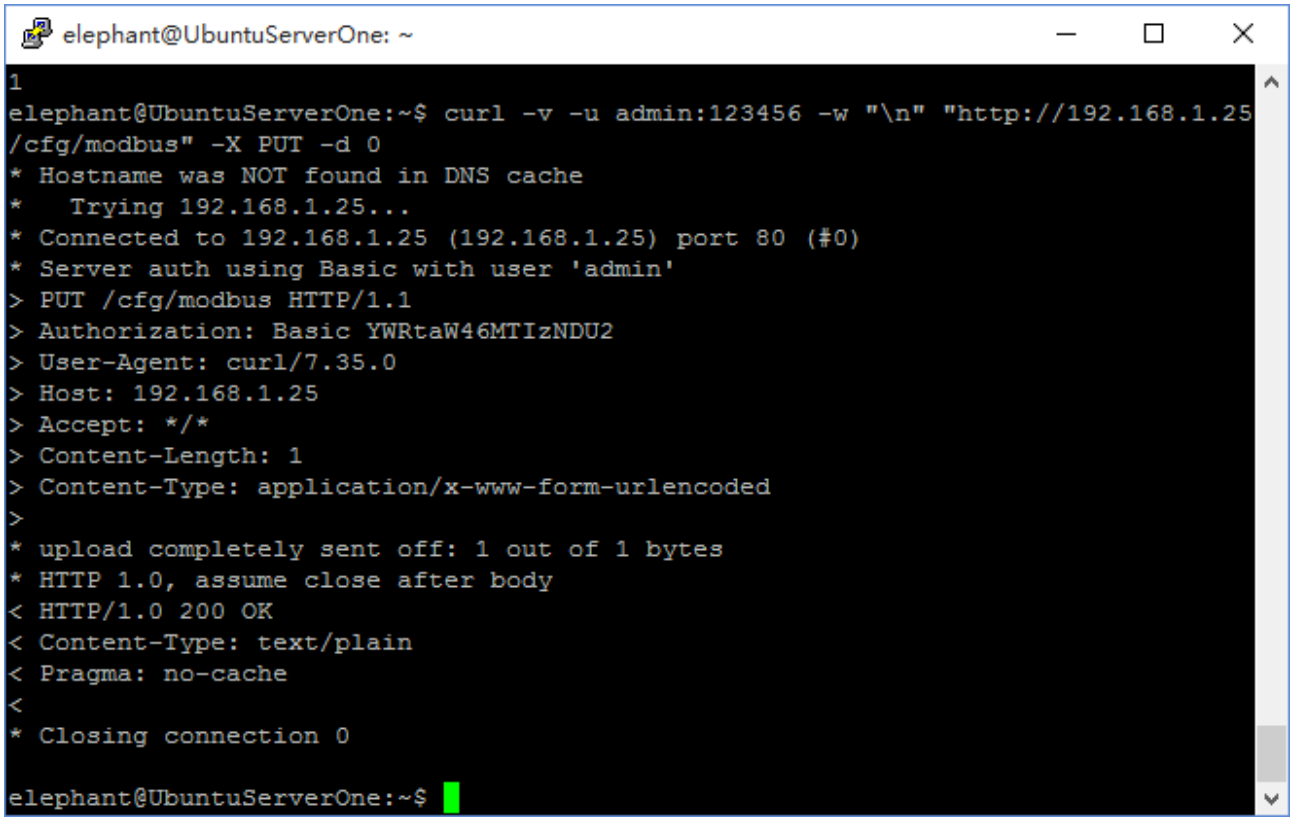
Modbus

资源路径：/cfg/modbus

url 示例：http://192.168.1.25/cfg/modbus
获取当前控制器的 modbus 配置



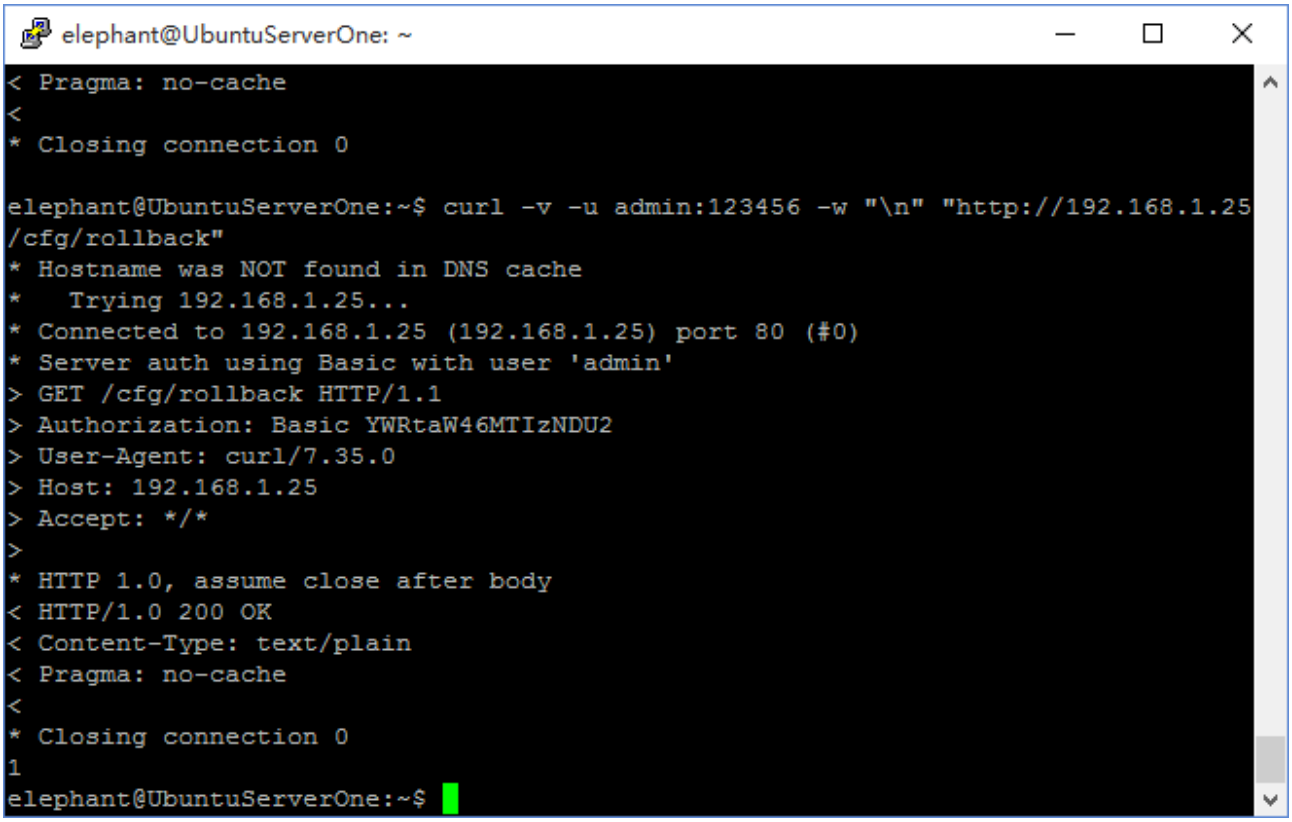
修改当前控制器的 modbus 为禁止



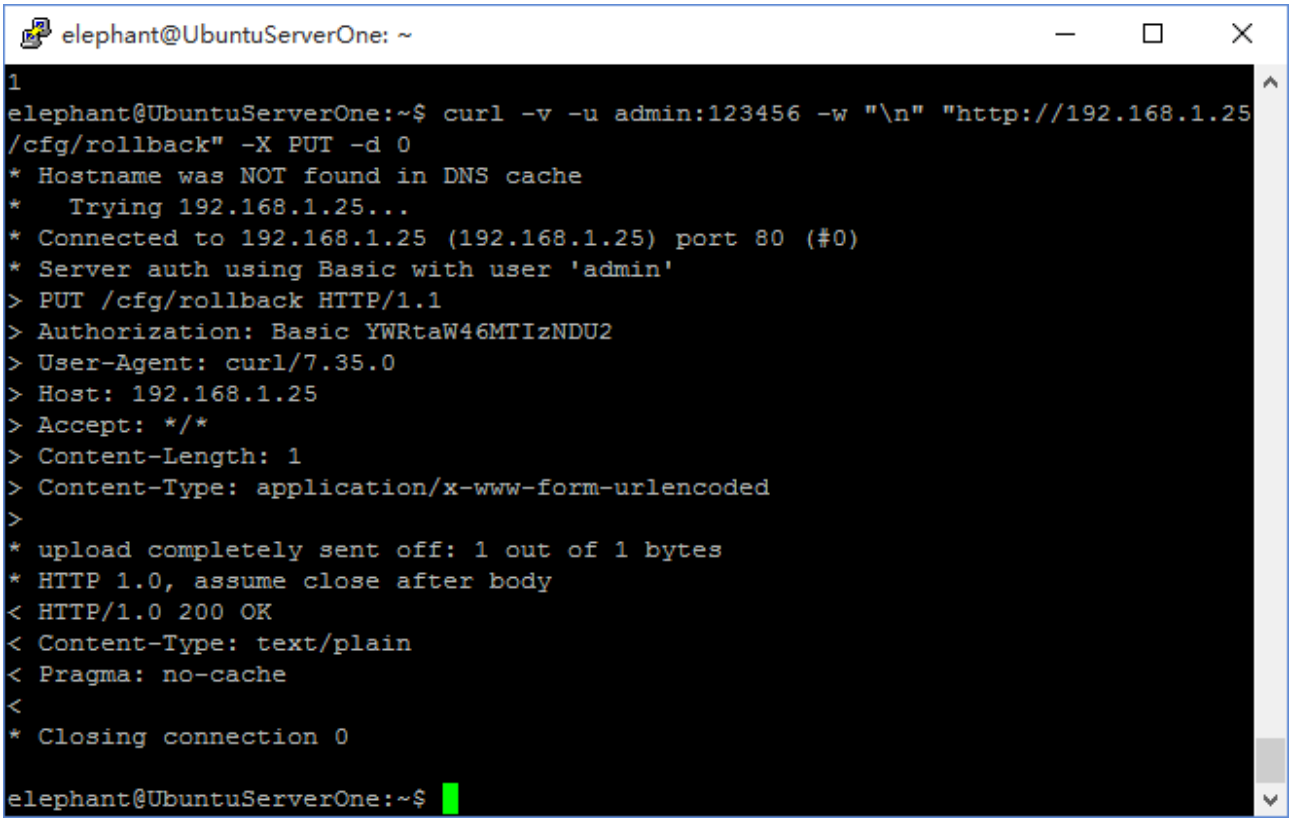
断电记忆

资源路径：/cfg/rollback

资源参数：1 或者 0,1 表示启用断电记忆，0 表示关闭。
url 示例：http://192.168.1.25/cfg/rollback
获取当前控制器的 rollback 参数

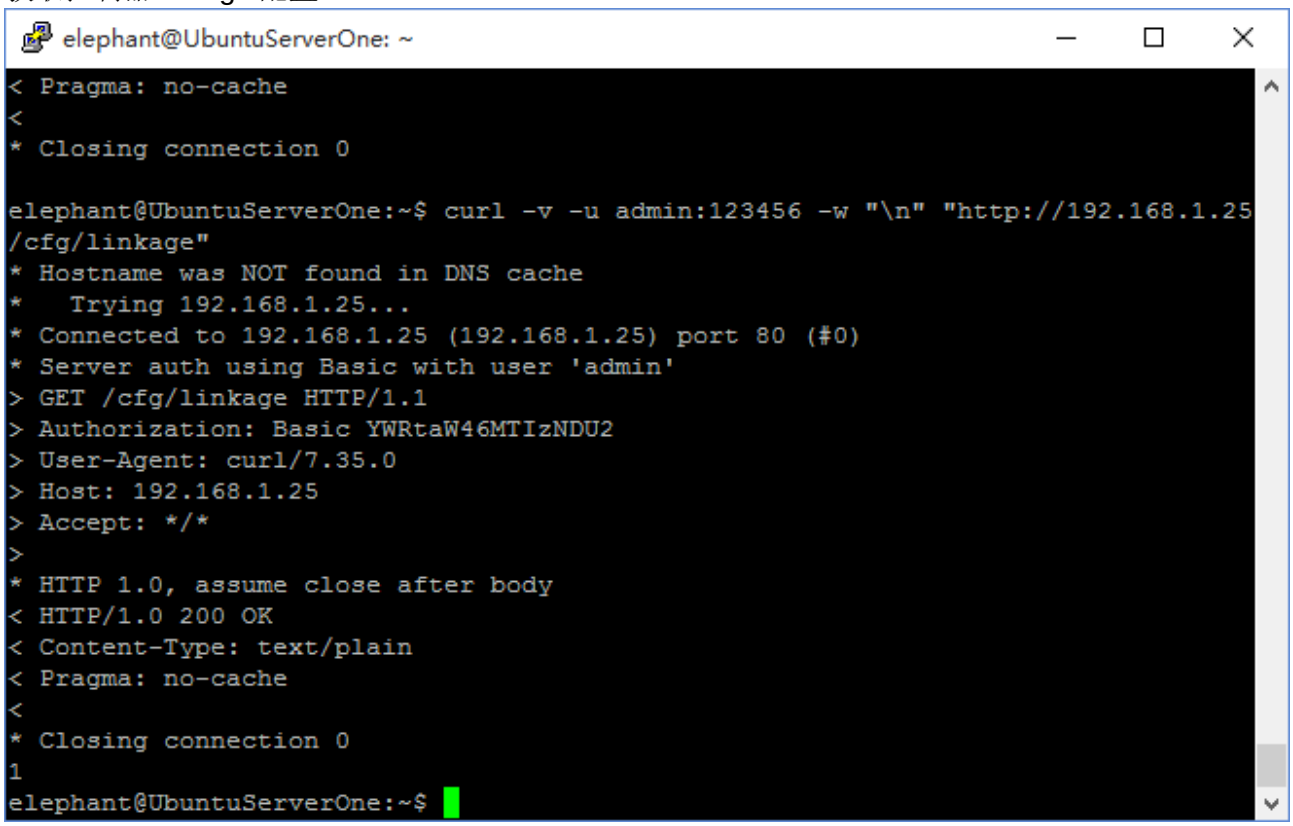


修改控制器断电记忆为关闭

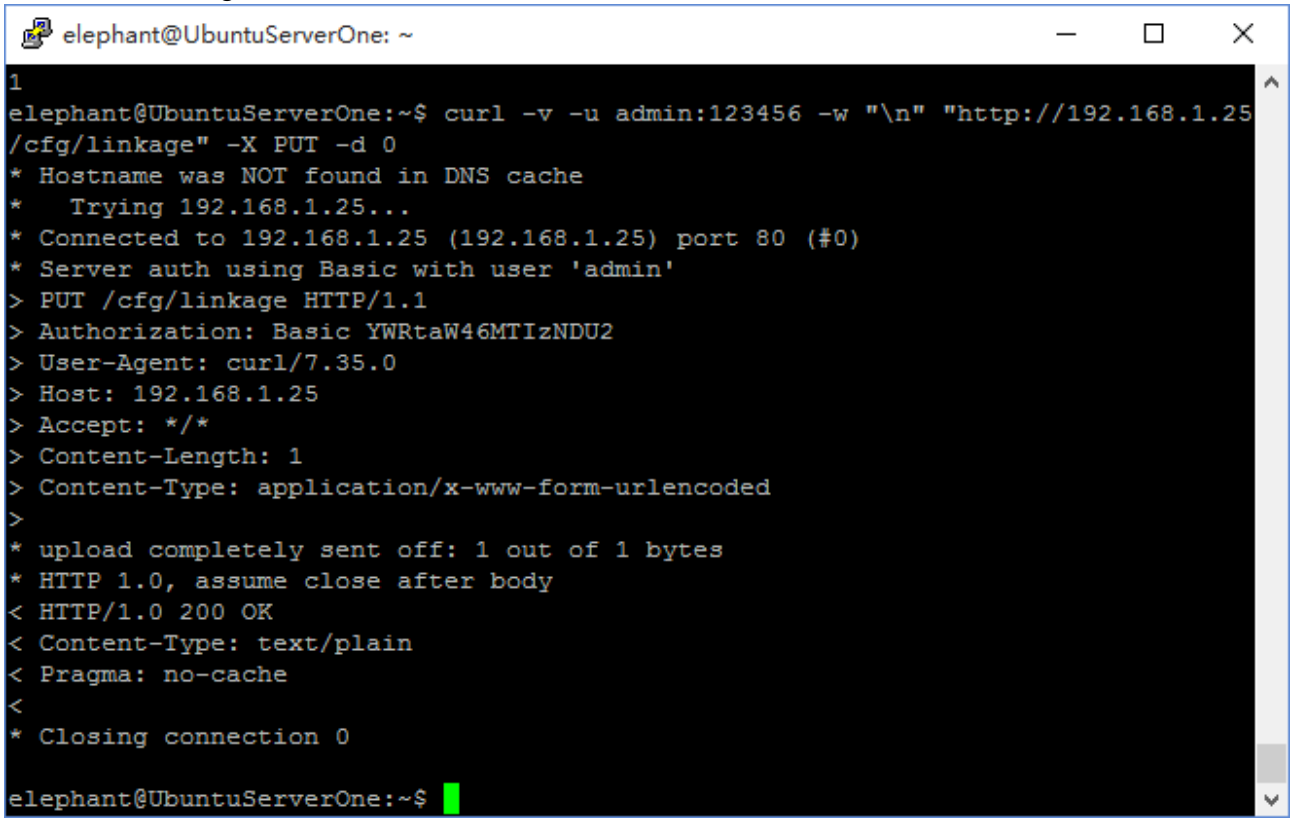


联动功能

资源路径：/cfg/linkage
资源参数：1 或者 0,1 表示启用联动，0 表示关闭联动
获取控制器 linkage 配置

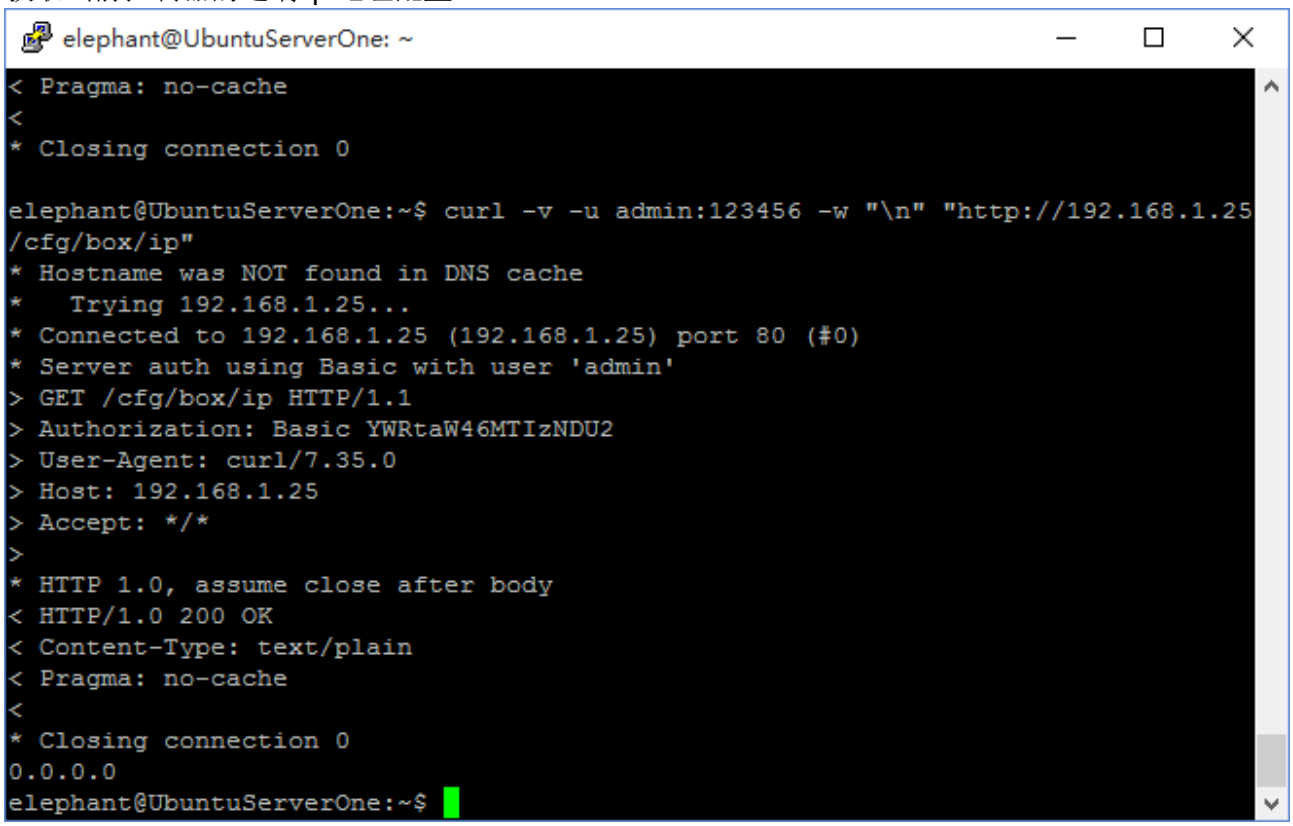


修改控制器 linkage 为关闭

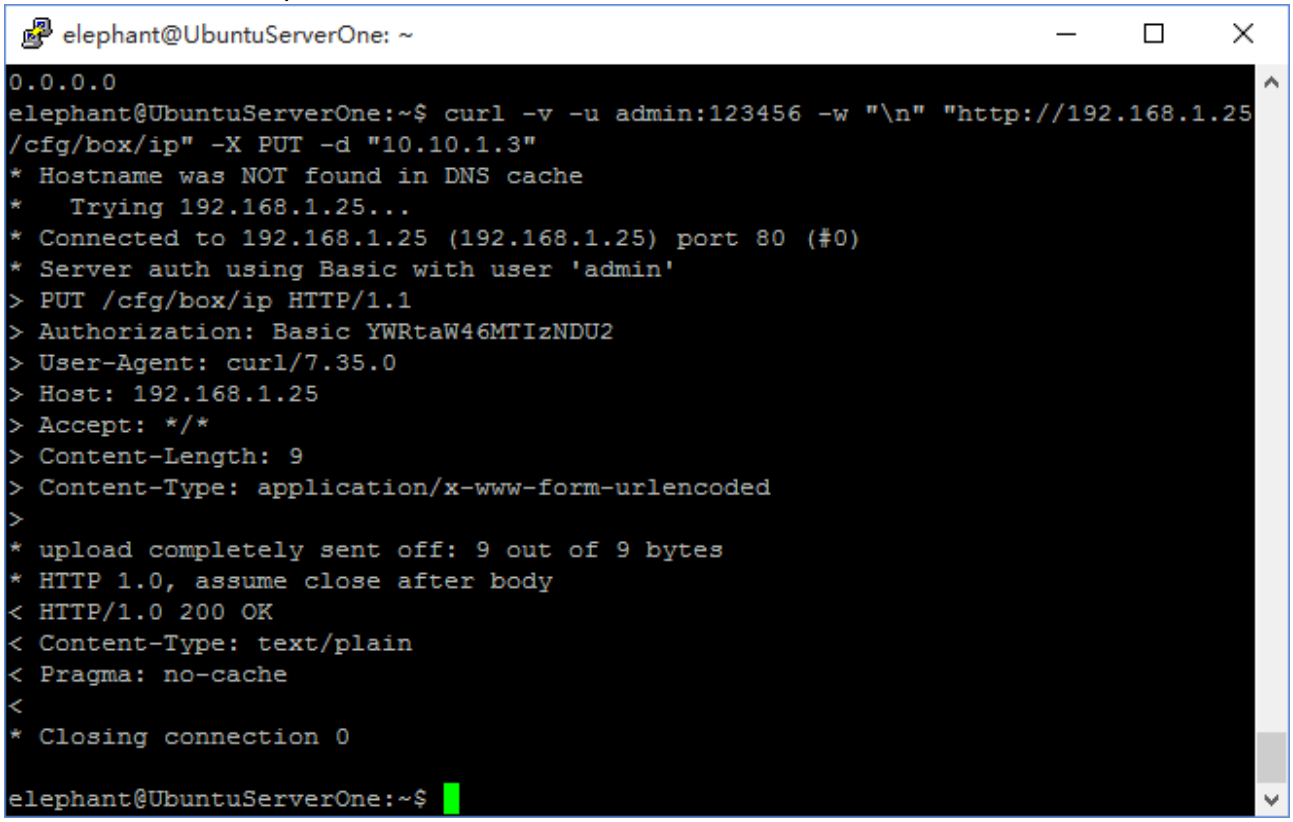


远端 IP

资源路径：/cfg/box/ip
url 示例：http://192.168.1.25/cfg/box/ip
获取当前控制器的远端 ip 地址配置

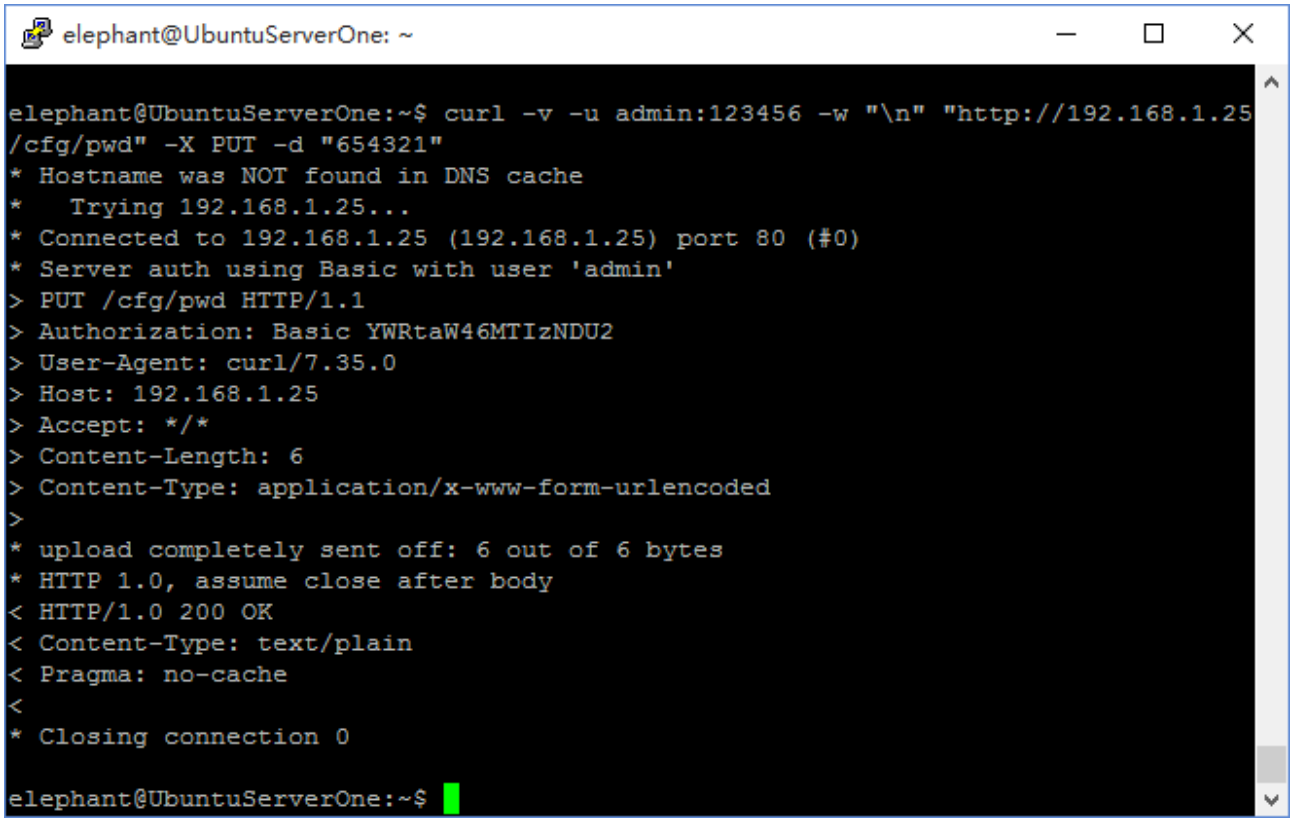


修当前控制器的远端 ip 地址为 10.10.1.3



密码

资源路径：/cfg/pwd
资源参数：字符和数字组成，最大为 6 个字符和数字的组合
修改控制器密码为 654321



MODBUS TCP

EIOC 系列支持读线圈（FC1）、读离散输入（FC2）、写线圈（FC5）、强制写多点线圈（FC15）四个功能码。关于 MODBUS-TCP 的详细信息请参考“开放型 Modbus/TCP 规范”，本手册不再做重复描述。为方便开发者，本手册对常用的命令进行示例说明。注意，以下示例中的数据为 16 进制。

读线圈

读取四个继电器的状态
发送：00 00 00 00 00 06 01 01 00 00 00 04
返回：00 00 00 00 00 04 01 01 01 0F
返回数据中的 0F（转换成二进制为 0b00001111）表示当前四个继电器的状态，最后一位为第一个继电器。

写线圈

将继电器 1 闭合
发送: 00 00 00 00 00 06 01 05 00 00 FF 00
返回: 00 00 00 00 00 06 01 05 00 00 FF 00
将继电器 2 闭合
发送 : 00 00 00 00 00 06 01 05 00 01 FF 00
返回 : 00 00 00 00 00 06 01 05 00 01 FF 00
将继电器 3 释放
发送 : 00 00 00 00 00 06 01 05 00 02 00 00
返回 : 00 00 00 00 00 06 01 05 00 02 00 00
将继电器 4 释放
发送 : 00 00 00 00 00 06 01 05 00 03 00 00
返回 : 00 00 00 00 00 06 01 05 00 03 00 00

强制写多点线圈

将四个继电器全部释放
发送 : 00 00 00 00 00 08 01 0F 00 00 00 04 01 00
返回 : 00 00 00 00 00 06 01 0F 00 00 00 04
将 1 号和 4 号继电器闭合，2 号和 3 号释放

发送 : 00 00 00 00 00 08 01 0F 00 00 00 04 01 09

返回 : 00 00 00 00 00 06 01 0F 00 00 00 04

发送数据中的 09（转换成二进制为 0b00001001）表示四个继电器的状态，最后一位为第一个继电器。

读离散输入

发送 : 00 00 00 00 00 06 01 02 00 00 00 04

返回 : 00 00 00 00 00 04 01 02 01 01

以上返回表示 INPUT1 闭合

发送 : 00 00 00 00 00 06 01 02 00 00 00 04

返回 : 00 00 00 00 00 04 01 02 01 09

以上返回表示 INPUT4 和 INPUT1 闭合，数据中的 09（转换成二进制为 0b00001001）表示四个开关量的状态，最后一位为 INPUT1。