**实 验 报 告**

|  |  |
| --- | --- |
| **课程名称：** | **物联网实验课** |
| **学生姓名：** | **张家健** |
| **学生学号：** | **202030482478** |
| **学生班级：** | **软件工程二班** |
| **开课学期：** | **大二下** |

**软件学院**

**2022 年 4 月**

# 任务一 开发ROS Sample App

## 实验目的

1. 熟悉Linux开发环境
2. 了解ROS SDK开发包结构
3. 掌握ROS SDK的App开发流程

## 实验环境

1. 硬件环境需求

1台R1511路由器

1. 软件环境需求

操作系统：Linux 虚拟机（推荐Ubuntu 18或20）

浏览器：推荐使用Chrome71以上

## 实验内容

### 准备Linux开发环境

准备Ubuntu的Linux开发环境，安装autoconf，cmake，vim等开发工具。

sudo apt update

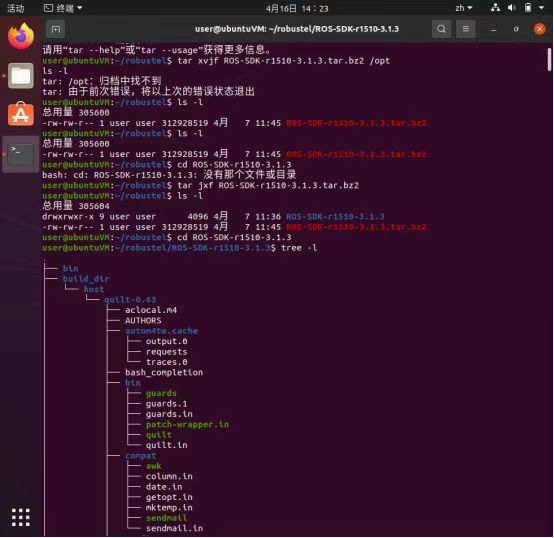
sudo apt install gcc make perl

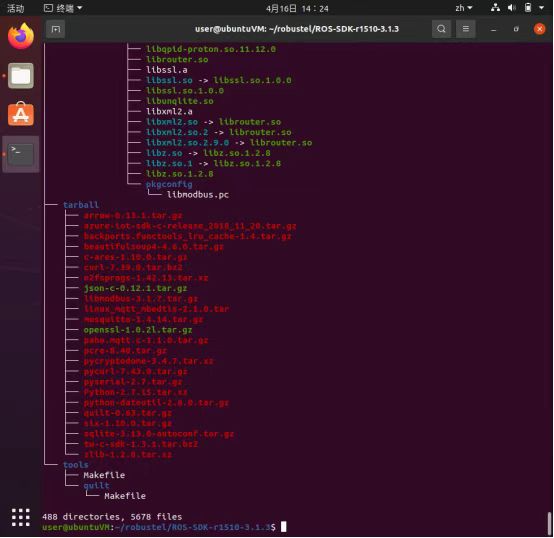
sudo apt install autoconf cmake vim tree g++ bison flex python ecj quilt zlib1g-dev pkg-config texinfo

### 安装ROS SDK开发包

通过WinSCP上传SDK开发包到Linux系统，运行 tar xvjf ROS-SDK-r1510-3.1.3.tar.bz2， 解压SDK包。

进入 ROS-SDK-r1510-3.1.3目录，输入tree . -L 2，查看SDK包目录结构。





### App源码目录结构说明

* package/xxx/Makefile

控制编译App工程的makefile文件，主要是填写App名字、描述、版本号等信息。同时负责准备编译前解压、复制源码和编译后的文件打包工作。

* Package/xxx/src/foo.c

App的源代码，支持多个c文件和h文件。

* Package/xxx/src/Makefile

控制编译App可执行程序的makefile文件，规则与通用的linux系统makefile相同。

* package/xxx/files/sdk.sh

路由器系统与App交互的接口，定义系统控制App运行状态的方法，以及系统不同运行阶段App对应的动作。

### 开发Hello world App

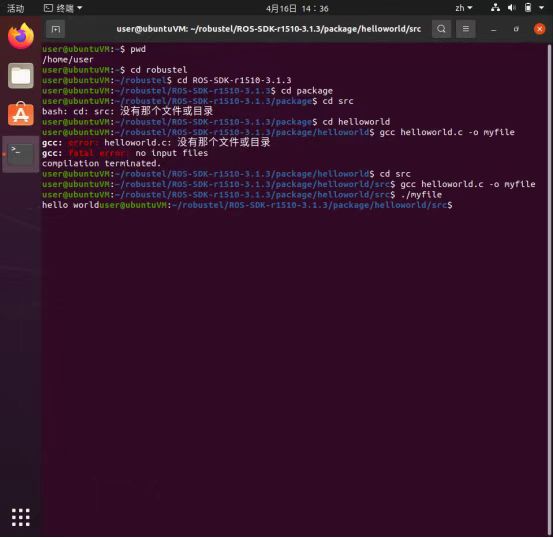
**创建helloworld目录**

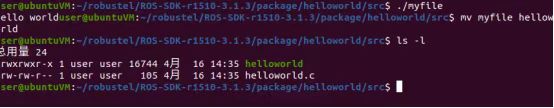
在package目录下新建 hellowworld目录，在helloworld目录下创建src和files目录。

这里通过mkdir指令来实现创建目录。

**添加源代码**

在package/helloworld/src目录下创建 helloworld.c文件，代码功能是通过syslog函数输出”hello world”打印信息。





一开始把c文件顺手命名为myfile，后面用mv指令来实现重命名。

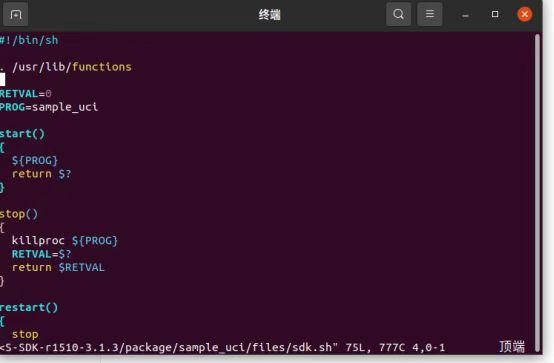
**添加源码Makefile文件**

在package/helloworld/src 目录下创建Makefile文件，实现将helloworld.c文件编译为helloworld 二进制文件。

**添加sdk.sh脚本**

参考package/sample\_uci/files/sdk.sh文件，在package/helloworld/files目录下新建sdk.sh文件。

PROG变量赋值改为helloworld。

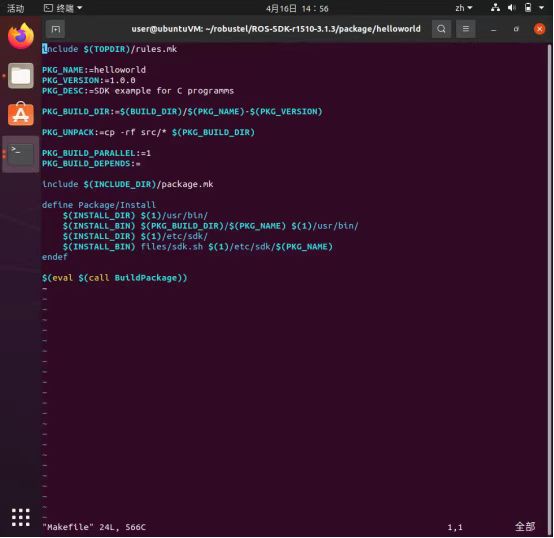


这里一开始重新给PROG变量赋值，后面重新创建这个文件的时候又忘记了赋值这一步，导致后面install那一步运行失败。

**添加工程Makefile文件**

参考package/sample\_uci/Makefile文件，在package/helloworld目录下新建Makefile文件。

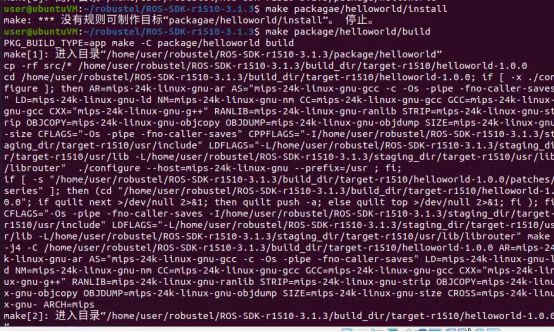
PKG\_NAME 赋值改为 helloworld，删除Package/Install下面拷贝uci.xml和usi.xml的步骤。



**编译helloworld**

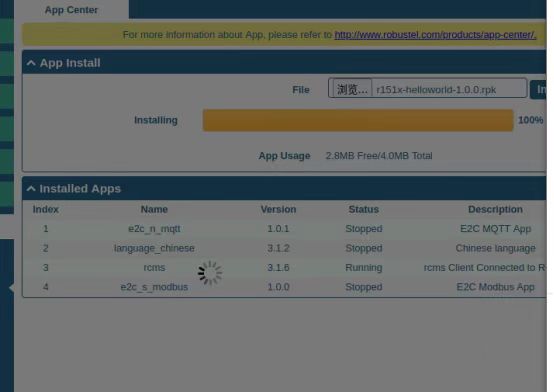
运行 make package/helloworld/clean 和 make package/helloworld/build 命令编译helloworld app。

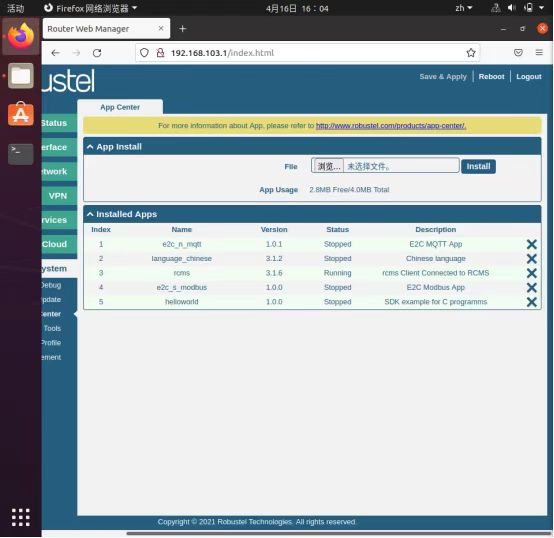
生成的安装包文件在bin/r1510/packages 目录下。



**安装App**

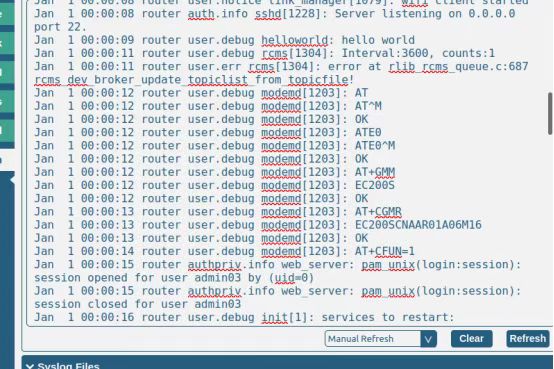
登录路由器管理界面，打开System -- App Center页面。选择编译生成的rpk文件，点击 “Install” 按钮进行安装，需要重启路由器才能生效。

****

****

**查看App 运行状态**

登录路由器管理界面，打开System -- Debug页面，查找 helloworld app 输出的打印信息。

****

## 实验小结

个人感觉这是整个任务书中最为困难的一步，在解压缩SDK包时就遇到了权限不足，解压方式不正确等问题，后面的文件编译也由于本机与示例中的目录和参数不同而出现了各种错误，总的来说还是因为对linux命令不熟悉导致消耗的时间比较长，在后续应该尽量多接触linux系统的使用场景，避免卡壳。

# 任务二 集成App到Web页面

## 实验目的

1. 熟悉ROS网页布局
2. 掌握常用的ROS UCI和USI属性
3. 掌握通过UCI设置程序参数
4. 掌握通过USI显示程序运行状态

## 实验环境

1. 硬件环境需求

1台R1511路由器

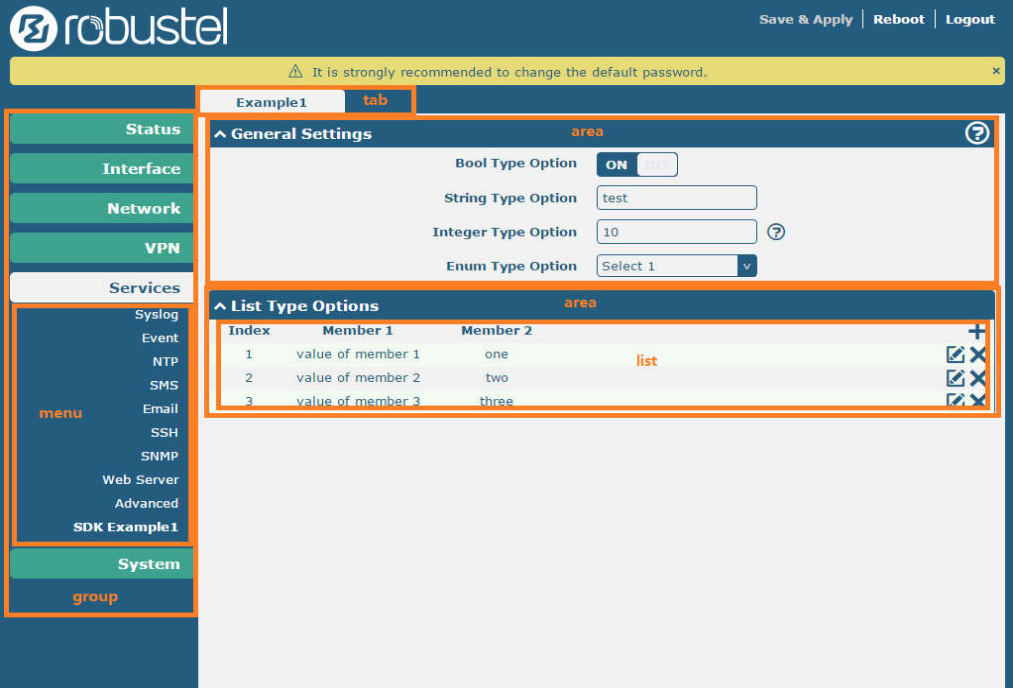
1. 软件环境需求

操作系统：Linux 虚拟机（推荐Ubuntu 18或20）

浏览器：推荐使用Chrome71以上

## 实验内容

### ROS Web页面布局



### UCI配置文件说明

* package/xxx/files/uci.xml

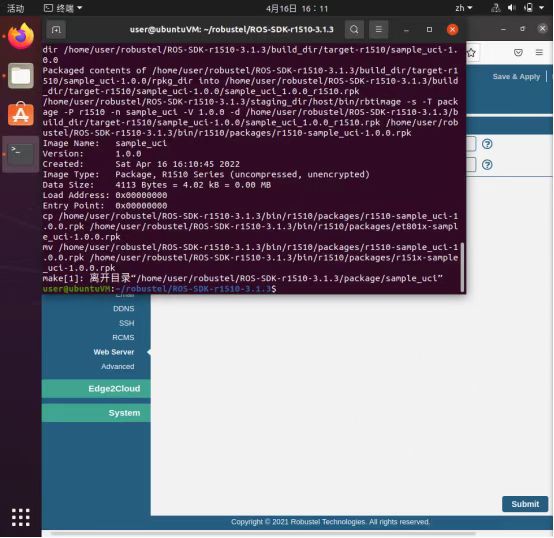
控制App输入参数的配置文件，在Web页面上输入的内容、属性，可在代码中通过uci\_get、uci\_get\_int等函数，获取到对应的输入内容。

* Package/xxx/files/usi.xml

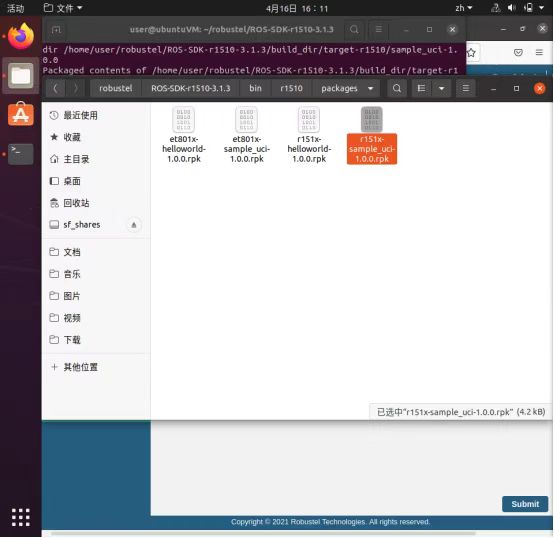
控制App显示运行状态的配置文件，通过在代码中调用usi\_clear、usi\_update、usi\_printf等函数，将App运行状态信息显示在网页上。

### 编译Sample UCI 工程

运行 make package/sample\_uci/clean 和 make package/sample\_uci/build 命令编译sample\_uci app。

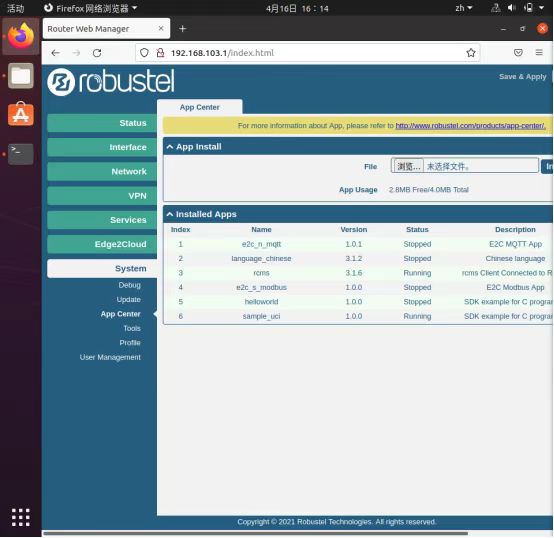


生成的安装包文件在bin/r1510/packages 目录下。

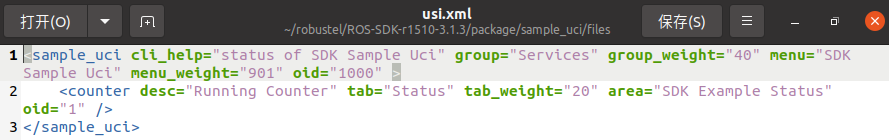
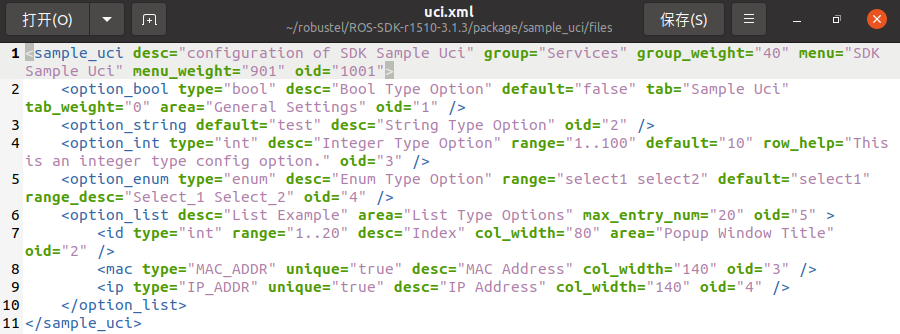
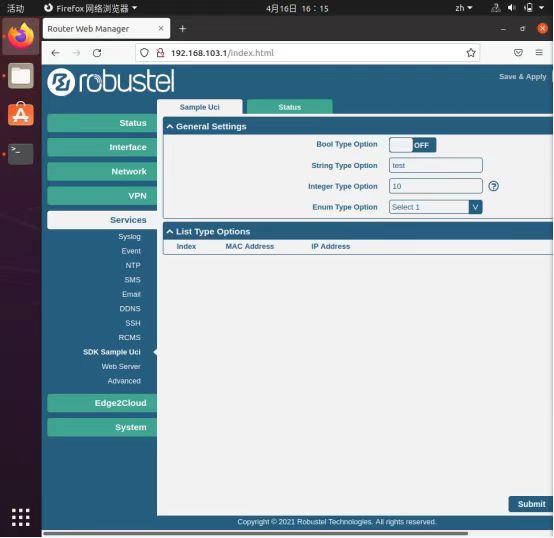


### 查看UCI及USI页面

登录R1511路由器管理页面，在System -- App Center页面安装编译sample\_uci生成的文件，并重启路由器。



在Services -- SDK Sample Uci 页面查看 “Sample Uci” 和 “Status” 页面，并和package/sample\_uci/files目录下的uci.xml 和usi.xml 做对比。



### 实验小结

这一步在编译Sample UCI 工程的时候花费了较长时间，对于其他内容还是比较容易执行。SDK提供了一种简单的方法，可以将自己的应用程序的配置和状态集成到路由器的Web管理器中。我们不需要了解任何Web编程语言，只需添加一个预定义格式的简单文件，网页就会自动生成，这里即集成App到Web页面。

# 任务三 通过Sample Modbus App采集Modbus传感器数据

## 实验目的

1. 熟悉Modbus传感器
2. 掌握R1511路由器通过RS485总线连接Modbus设备
3. 掌握ROS 下Modbus采集App的开发

## 实验环境

1. 硬件环境需求

1台R1511路由器，Modbus传感器，专用485连接线，网线

1. 软件环境需求

操作系统：Linux 虚拟机（推荐Ubuntu 18或20）

浏览器：推荐使用Chrome71以上

## 实验内容

### 连接示意图



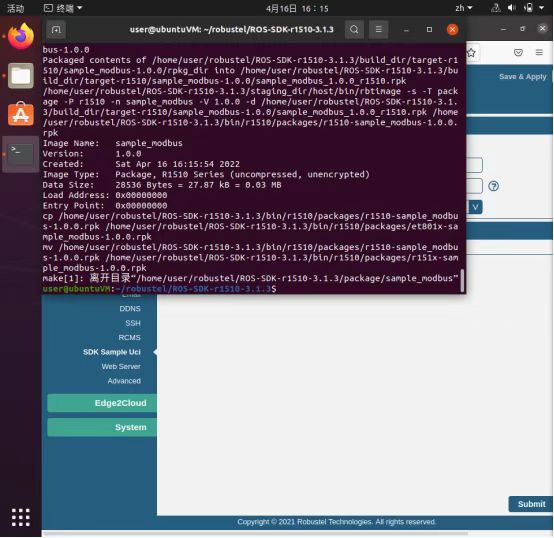


### 传感器通讯协议

查看传感器使用手册，在“寄存器地址及数据转换方式”一节查找采集数据保存的寄存器地址，

### 编译Sample Modbus

运行 make package/sample\_modbus/clean 和 make package/sample\_modbus/build 命令编译sample\_modbus app。



生成的安装包文件在bin/r1510/packages 目录下。

### 配置UCI参数

打开 Edge2Cloud -- Sample Modbus页面，设置Sample Modbus的UCI参数。

Enable：是否启用Sample Modbus App；

Read Interval：采集间隔，单位秒；

Sensor Tag：采集信息的标签，自定义；

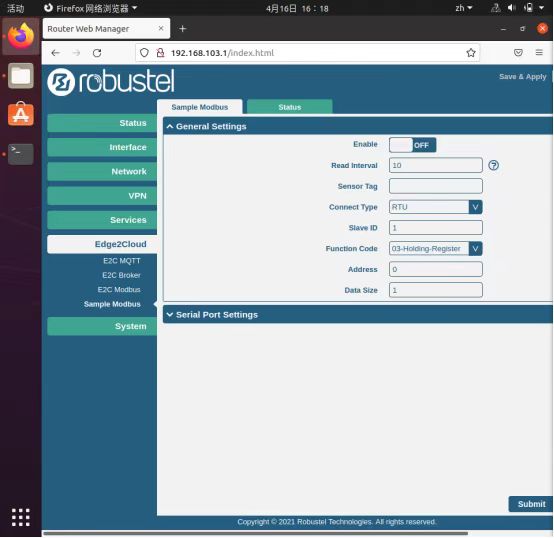
Connect Type：连接类型；

Slave ID：Modbus子设备地址；

Function Code：Modbus 协议功能码；

Address：Modbus设备数据寄存器地址；

Data Size：Modbus数据长度，1个寄存器数据长度为两个字节；

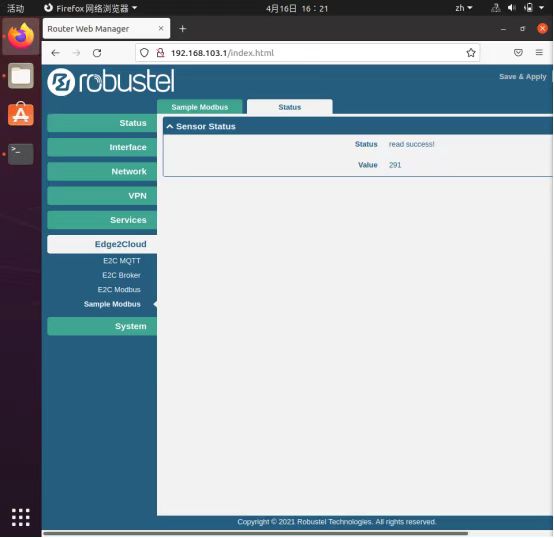


### 查看采集信息

打开 Edge2Cloud -- Sample Modbus页面，切换到Status标签。

Status：Modbus数据是否采集成功；

Value：Modbus寄存器的值；



### 实验小结

这一步属于采集传感器所收集的信息，实时地传输给APP。