原创 noevil 于 2023-09-20 17:36:42 发布 92 收藏 版权
文章标签： linux 运维 服务器 c++

Onvif是没有开源库的，在C++上开发onvif相关功能必须使用gSOAP工具根据wsdl文件来生成相关源代码文件。这一过程有些繁琐，但是也带了了很多灵活性——功能是自定义的。

编译gSOAP

下载

gSoap分商业版和开源版本，开源版可以从gSOAP Toolkit download | SourceForge.net下载

要用gsoap生成onvif源码，必须用到wsdl2h和soapcpp2两个工具（执行文件）。

wsdl2h默认不支持HTTPS，编译的时候需要开启https支持——默认编译支持。

也就是说我们最好从源代码去编译安装，然后就可以直接使用https的在线wsdl文档了。

安装依赖

gSOAP有三个依赖需要安装

```
sudo apt install bison flex openssl
```

- OpenSSL (3.0 or 1.1, earlier versions are supported but not recommended) or GNUTLS and the Zlib libraries to enable SSL (HTTPS) and compression. These libraries are available for most platforms and are often already installed.
- Flex http://flex.sourceforge.net and Bison Bison- GNU Project - Free Software Foundation to build the soapcpp2 tool. You can also build soapcpp2 without Bison and Flex installed, see Genivia Product Downloads for details.

也可以从上面地址去下载自己编译安装依赖，甚至不装。

编译gSOAP

找个目录解压刚刚下载的gSOAP源码包，进入目录然后编译安装。

```
> ./configure
> make -j4
> sudo make install
```

根据wsdl文档编译onvif源代码

这里的所有步骤我已经加入了开源项目https://github.com/NoevilMe/onvif_demo.git

也有比较详细的文档说明。

先确定一个目录，这里不是gsoap的源代码目录，是用来写测试代码的地方，名字任取吧。

复制会用到的源代码

在确定的工作目录下，创建一些子目录用来进行后续的工作

```
.
├── gsoap
│   ├── custom
│   ├── import
│   └── plugin
├── onvif
├── onvif_head
├── soap
└── tests
```

或者可以将下面的代码保存成 copy_from_source.sh ，然后运行。注意其中SRC_DIR必须修改成你自己的解压出来的包目录。

```
# 将要用到的一些源码文件复制到gsoap目录

# 自己修改源码目录
SRC_DIR=~/opensource/gsoap-2.8

if [ ! -d $SRC_DIR ]; then
    echo ${SRC_DIR} does not exist!
    exit 1
fi

FROM_DIR=$SRC_DIR/gsoap
if [ ! -d $FROM_DIR ]; then
    echo ${FROM_DIR} does not exist!
    exit 1
fi

if [ ! -d gsoap ]; then
    mkdir gsoap
fi

TARGET_DIR=gsoap

# 要复制的目录和文件
LISTS="custom import plugin dom.cpp stdsoap2.h stdsoap2.cpp"
for t in ${LISTS}; do
    ft=${FROM_DIR}/${t}
    if [ -d ${ft} ]; then
        echo ${ft} is an directory
        cp -rf ${ft} ${TARGET_DIR}
    elif [ -f ${ft} ]; then
        echo ${ft} is a file
        cp -rf ${ft} ${TARGET_DIR}
    else
        echo ${ft} does not exist!
    fi
done

# typemap.dat 不要覆盖，这个会被修改了
if [ ! -f gsoap/typemap.dat ]; then
    cp ${FROM_DIR}/typemap.dat gsoap/typemap.dat
else
    echo "typemap.dat exists! do not copy it!"
fi
```

这一步会将一些生成onvif源码过程中用到的gsoap源代码复制到gsoap目录下

分类专栏
Linux C++ 1篇

- gsoap/import
- gsoap/custom
- gsoap/plugin
- gsoap/stdsoap2.cpp
- gsoap/stdsoap2.h
- gsoap/typemap.dat
- ...

修改typemap.dat

由于后续编译过程中需要用到 duration.c 文件，会遇到类型LONG64报错的问题，需要gsoap/typemap.dat 文件中取消以下行的注释：

```
xsd__duration = #import “custom/duration.h” | xsd__duration
```

生成头文件onvif.h

执行命令

```
./step1_gen_head.sh
```

此步骤会生成onvif_head/onvif.h文件

该脚本会在线下载wsdl文件（需要自己配置），并且修改onvif.h文件，加入鉴权的相关项。

命令解析

step1_gen_head.sh主要使用了wsdl2h命令来生成onvif.h文件。wsdl2h参数解析：

```
-c： 生成c风格代码（注：后缀名还是.cpp，但实际上是.c）
-c++：生成c++风格代码（注：默认是生成c++代码）
-x：表示不生成xml 文件（注：生成的xml文件，有助于了解发送是SOAP是怎样的结构，建议不使用-x）
-l：表示指定导入路径
-C：表示生成客户端代码
-S：表示生成服务端代码
-s：不使用STL代码
-o: 生成.h文件叫什么名字
-t：后面紧跟“typemap.dat”这个批处理文件
```

The wsdl2h tool performs the mapping of WSDL and XML schemas to C and/or C++ automatically. The output of wsdl2h is a "data binding interface file" which is simply an annotated C/C++ header file with the serializable C/C++ data types that represent XML schema components. This file also includes comments and documentation of the serializable data types.

生成的onvif.h是一个数据绑定接口定义文件，后续步骤会用到。

关于鉴权

鉴权也就是设备使用用户名密码登录。

如果onvif.h不加入#import "wsse.h"，使用soap_wsse_add_UsernameTokenDigest函数会导致编译出错，也就无法登录设备进行操作了。

但是默认生成的onvif.h中是没有#import "wsse.h"的。step1_gen_head.sh脚本已经处理了这个问题。

```
#加入鉴权，发送请求需要用户名和密码
sed -i '122 a #import "wsse.h"' ${DST}
```

wsdl的相关功能描述

- - https://www.onvif.org/ver10/device/wsdl/devicemgmt.wsdl 用于获取设备参数
- - https://www.onvif.org/onvif/ver10/network/wsdl/remotediscovery.wsdl 用于发现设备
- - https://www.onvif.org/onvif/ver20/ptz/wsdl/ptz.wsdl 云台控制
- - https://www.onvif.org/onvif/ver10/media/wsdl/media.wsdl 获取264的视频流地址
- - https://www.onvif.org/onvif/ver20/media/wsdl/media.wsdl 获取h265视频流地址
- - http://www.onvif.org/onvif/ver20/imaging/wsdl/imaging.wsdl 光圈，对比度，饱和度

更多的我暂时没用到，也就不举例了。

SOAP_ENV__Fault重复定义

如果没有修改相关文件，生成代码的时候会出现如下错误。

```
wsa5.h(280): *WARNING*: Duplicate declaration of 'SOAP_ENV__Fault' (already declared at line 268)
wsa5.h(290): **ERROR**: service operation name clash: struct/class 'SOAP_ENV__Fault' already declared at wsa.h:278
```

之所有会出现这个错误，是因为onvif.h头文件中同时：

```
#import "wsdd10.h" // wsdd10.h中又#import "wsa.h"
#import "wsa5.h"   // wsa.h和wsa5.h两个文件重复定义了int SOAP_ENV__Fault
```

解决方法：

修改import\wsa5.h文件，将int SOAP_ENV__Fault修改为不冲突的任何名字，例如int SOAP_ENV__Fault_xxx，再次使用soapcpp2工具编译就成功了。

脚本也已经自动处理了，具体实现是

```
sed -i 's/int SOAP_ENV__Fault$/int SOAP_ENV__Fault_xxx/g' gsoap/import/wsa5.h
```

生成onvif相关源代码

执行生成命令

```
./step2_gen_code.sh
```

脚本已经删除了一些无用文件、复制并重命名了相关文件。其中onvif.h文件其实已经没用了，可以删掉，不需要参与后续IPC客户端程序的编译。这里有好多个命名空间的.nsmap文件，文件内容都一模一样，拿wsdd.nsmap一个来用即可。

soap目录

这里把从gsoap中将来用到的一些文件复制过来了，修改成了cpp后缀

onvif目录

一些生成的源代码文件，就是我们要的东西。

- 各种nsmap文件：命名空间，除了名字不一样，内容是一样的，里面的内容竟然是每一个xml文件里的Envelope字段内容。我们只需要留下一个就可以了，并将之改名为wsdd.nsmap

- soapC.cpp：指定数据结构的序列化和反序列化
- soapClient.cpp：客户端代码
- soapH.h：主头文件，所有客户机和服务器源代码都要包括它
- soapStub.h：从输入头文件（onvif.h）生成的经过修改且带命名空间前缀的头文件

示例

还是参考工程NoevilMe/onvif_demo: Linux c++ onvif client demo (github.com)

CMakeLists

```
cmake_minimum_required(VERSION 3.0)
project(OnvifSoap)

set(CMAKE_CXX_STANDARD 11)

set(CMAKE_CXX_FLAGS "-g -O0")

set(LIB_SOAP_SRC
    soap/struct_timeval.cpp
    soap/duration.cpp
    soap/wsaapi.cpp
    soap/dom.cpp
    soap/wsseapi.cpp
    soap/smdevp.cpp
```

soap目录编译成了libonvif_soap,依赖openssl。 还需要定义两个宏WITH_OPENSSL和 WITH_DOM

onvif目录编译成了libonvif。 依赖libonvif_soap.

demo

位于tests/scan_device.cpp

```
#include "onvif/soapH.h"
#include "soap/wsaapi.h"
#include <assert.h>
#include <stdio.h>
#include <stdlib.h>

#include "onvif/wsdd.nsmap"

#define SOAP_ASSERT assert
#define SOAP_DBGLOG printf
#define SOAP_DBGERR printf

#define SOAP_SOCK_TIMEOUT (10) // socket超时时间（单秒秒）
```

执行结果

```
$ ./dev_scan
http://10.10.10.103/onvif/device_service, tdn:NetworkVideoTransmitter tds:Device
http://10.10.10.104/onvif/device_service, tdn:NetworkVideoTransmitter tds:Device

detect end! It has detected 2 devices!
```

tdn:NetworkVideoTransmitter

> NVT (Network Video Transmitter)
>
> ```
> A Network Video Transmitter (NVT) is an ONVIF device that sends media data over
> an IP network to a client. For example, an NVT may be an IP network camera or
> an encoder device.
> ```
>
> An NVT implements the following services to provide its core functionality:
>
> - *Device service* enables an NVT to provide device management functionality such as device capabilities, system and network settings, security settings and firmware upgrade.
> - *Event service* enables an NVT to send events to clients. Media service enables an NVT to stream media data to clients. Media data includes video, audio, video analytics and other metadata.
> - *Device IO service* enables an NVT to support physical inputs and outputs. An NVT can also implement the following services to provide extended functionality:
> - *PTZ service* enables an NVT to provide PTZ control if the device is a PTZ camera.
> - *Imaging service* enables an NVT to provide configuration of image settings which affect the visual appearance of the video, for example, exposure time, gain and white balance, focus control.
> - *Video Analytics service* enables an NVT to provide video analytics functionality.
>
> Beyond this, an NVT can also include additional ONVIF services, for example the Recording service if support for local storage is required

网络视频服务器(比如，网络摄像机，编码设备等)，通过IP网络发送媒体数据到客户端。

私信 关注

搜博主文章

热门文章

Linux c++ onvif客户端开发(1): 根据wsdl生成cpp源文件 91

Linux C++编程之Eventloop 19

H.264视频码流NALU分割 7

最新评论

Linux c++ onvif客户端开发(1): 根据wsdl…
CSDN-Ada助手: 恭喜您写了第三篇博客，标题看起来非常有趣！通过根据wsdl生成…

Linux C++编程之Eventloop
CSDN-Ada助手: 恭喜您开始博客创作！标题"Linux C++编程之Eventloop"非常吸引…

Linux C++编程之Eventloop
CSDN-Ada助手: 恭喜你这篇博客进入【CSDN每天最佳新人】榜单，全部的排名请看…

您愿意向朋友推荐“博客详情页”吗?

强烈不推荐 不推荐 一般般 推荐 强烈推荐

最新文章

H.264视频码流NALU分割

Linux C++编程之Eventloop

2023年 3篇

Linux下gdb常规调试 最新发布 — Bit_Dong的博客 105
gdb全称"GNU symbolic debugger"，从名称上不难看出，它诞生于 GNU 计划（同时诞生的还有 GCC、Emacs 等），是 Linux 下常用的程序调试器。发展…

转载—Linux下文件搜索、查找、查看命令 — qq_21952195的博客 58
Linux下文件搜索、查找、查看命令

Linux下库的入门与制作 — weixin_46216674的博客 190
Linux下库的入门与制作

Linux 的性能调优的思路 — qq_48811377的博客 166
系统性能优化是个涉及面广、繁琐、长久的工作，寻找出现性能问题的根源往往是最难的部分，一旦找到出现问题的原因，性能问题也就迎刃而解。因此…

Linux——vi编辑器 — weixin_68256171的博客 111
1、最早可追随到1991年，全称为"Vi IMproved"2、模式——命令模式——末行模式——编辑模式3、使用vi/vim命令编辑文件——在每次运行vim编辑器时…

linux-第三章-软硬链接区别 — m0_72210904的博客 220
硬链接：多个文件名指向同一个索引（inode）节点号作用：防止误删除（备份的是文件名，并不是文件数据），节省磁盘的大量空间对象：文本文件，不…

Linux--信号 — weixin_48208102的博客 321
比如，终端用户输入了ctrl+c来中断程序，会通过信号机制停止一个程序。需要高速内核，用户希望如何处理某一种信号，说白了就是写一个信号处理函数…

linux开机自动启动java的jar包项目及开机自动启动Nacos的配置 — 老照片 131
linux开机自动启动java的jar包项目及开机自动启动Nacos的配置

Linux的调试工具 - gdb(超详细) — originalHSL的博客 883
本文详细介绍了Linux的调试工具gdb,演示了gdb主要使用的调试指令。

Linux怎么查看group — m0_61629312的博客 168
2023年9月20日，周三晚上。

Linux备份策略：保证数据安全 — 授人以鱼不如授人以渔，知之者不如好之者，好之… 183
Linux基础 Shell脚本 Linux命令 运维自动化 Docker容器 Kubernetes Linux安全 Nginx配置 Apache部署 CI/CD流程 Ansible自动化 Linux网络配置 RAID配置…

Linux从root账号切换到普通账号并执行shell脚本 — 纸上得来终觉浅，绝知此事要躬行 174
su es -s /bin/bash _start_es.sh 脚本自动切换账号并执行其他脚本

进程同步与互斥 — 淳潜 的博客 145
1.P、V操作必须成对出现2.互斥操作时，P、V操作出现在同一进程3.同步操作时，P、V操作出现在不同进程4.既有同步、又有互斥操作时，同步信号量P…

C++ onvif客户端下载 — 06-13
您可以访问ONVIF的官方网站（https://www.onvif.org/）获取ONVIF协议的相关文档和标准。如果您想要使用C++编写ONVIF客户端，可以使用一些第三方…

“相关推荐”对你有帮助么?

非常没帮助 没帮助 一般 有帮助 非常有帮助

noevil 关注

2 0 1

举报