**CVNWARE平台**

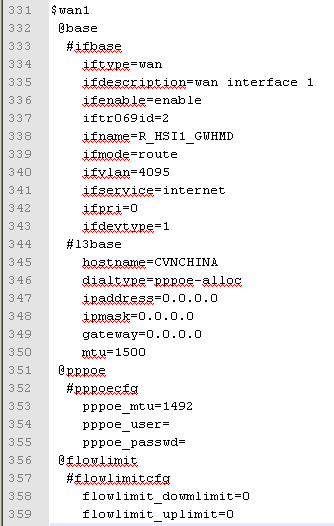
**接口管理详细介绍**

# 简介

CVNWARE平台中的接口管理模块是比较重要的一个模块，想要很轻松地在CVNWARE平台上进行相关特性的开发，熟悉接口管理模块是必要的。本文不介绍接口管理的实现原理，因为这些太过理论，而且已经有相关的文档可以参考，而是重点介绍如何利用接口管理进行相关特性的开发，用于指导涉及到CVNWARE平台移植工作的开发人员。

# 添加接口配置

首先需要在系统的默认配置文件startup（位于config文件夹下）中添加该接口相关配置，如下图所示：



#ifbase表示接口的基本属性，以下为具体解释：

1. iftype表示接口类型，如果是wan接口，则为wan；如果为桥接口，则为br，如果为无线口，则为wlan。
2. ifdescription表示接口描述，可以任意填写；
3. ifenable表示接口使能，enable表示使能，disable表示去使能；
4. ifname表示接口的伪名，可以手动配置；
5. ifmode表示接口的工作模式，route表示路由模式，bridge表示桥接模式；
6. ifvlan表示接口的VLAN ID；
7. ifpri表示接口的优先级；
8. ifservice表示接口的服务类型；
9. iftr069id表示TR069 ID，TR069模块需要通过它来获取对应接口的索引；
10. ifdevtype表示接口的设备类型，这个字段是新增字段，主要是为了区分不同的WAN接口，因为WAN接口可能是以太类型的WAN接口， 也可能是3G口或Ap-Client口。

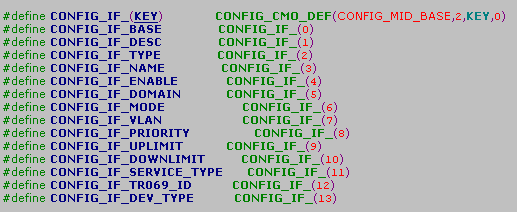
#l3base表示接口的三层属性，以下为具体解释：

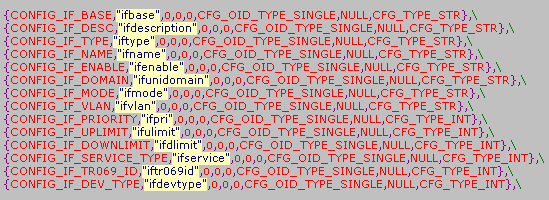
1. hostname表示主机名，只有WAN口的拨号方式为dhcp时才有用；
2. dialtype表示接口的拨号方式，static表示静态地址方式，dhcp-alloc表示DHCP获取方式；pppoe-alloc表示PPPoE获取方式；
3. ipaddress表示接口IP地址；
4. ipmask表示接口的子网掩码；
5. gateway表示接口的默认网关；
6. mtu表示接口的MTU值。

@pppoe部分表示PPPoE相关配置信息，pppoe\_user表示PPPoE用户名，pppoe\_passwd表示PPPoE密码，pppoe\_mtu表示PPPoE的MTU和MRU值。

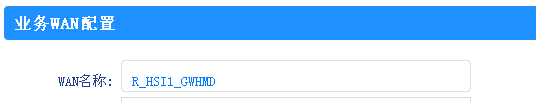
@flowlimit部分表示接口的限速相关配置信息。

文件startup中添加完毕之后，还需要修改头文件root/ platform/include/ lw\_config\_oid.h，添加对应的配置项，这些是配置管理的内容，这里只是简单提一下，不细究，如下图所示：





注意：上图中，startup文件第331行的$wan1表示该接口的ifname为wan1，第338行的ifname=R\_HSI1\_GWHMD表示其伪名为R\_HSI1\_GWHMD，其中，ifname是无法修改的，而伪名则可以在页面上修改：

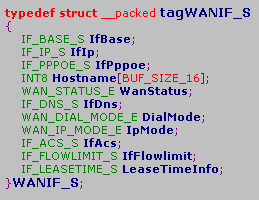


# 添加代码

假设工程根目录为root，并以添加WAN接口为例。

## 创建对应头文件

创建一个头文件lw\_if\_wan.h，主要是定义WAN接口的信息结构体，如下图所示：



上图的结构体是可以跟startup文件中的wan口相关配置对应起来的，比如IfBase表示接口基本属性信息，IfIp为地址信息，IfPppoe为PPPoE信息，Hostname为主机名，WanStatus为WAN接口状态，IfDns为DNS信息，DialMode为拨号方式，IpMode为IP地址类型，IfAcs为ACS服务器信息，IfFlowlimit为接口限速信息，LeaseTimeInfo为租约时间信息。

## 修改文件vos\_config.c

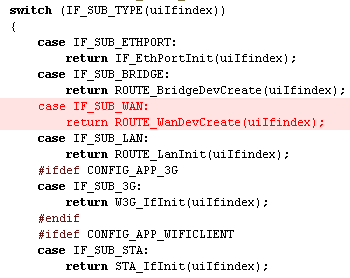
在文件root/platform/cfgm/newcfg/vos\_config.c中的lwConfigSetUpForEachIf函数中添加如下语句：



其中宏IF\_WAN\_NAMEPREFIX的值为“wan”，表示wan接口。

添加上面的代码后，系统在接口初始化的时候就会读取默认配置文件startup的$wan1的相关配置，并根据这些配置，通过函数IF\_CreatInterface创建该接口，函数IF\_CreatInterface执行完毕后，该接口的索引就会被创建，即上图中的ifindex，后续的很多操作都是通过接口索引进行的。

其中IF\_CreatInterface会调用到函数IF\_InitInterface，在该函数中需要添加以下红色字体的语句，用于完成WAN接口初始化相关操作，包括创建虚接口，设置IF\_LINKNAME和IF\_STATE等。

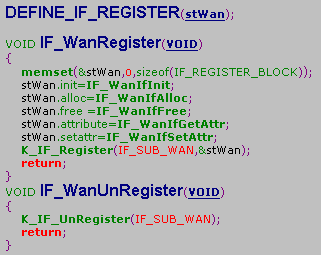


## 添加接口管理相关文件

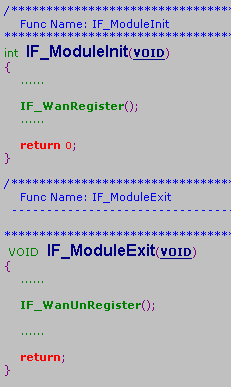
在root/ifm/ifm下创建文件ker\_if\_wan.c和ker\_if\_wanif.c，这两个文件就是WAN接口管理文件。

同理，如果是无线口，则为ker\_if\_wlan.c和ker\_if\_wlanif.c；如果是3G口，则为ker\_if\_3g.c和ker\_if\_3gif.c；如果是STA口，则为ker\_if\_sta.c和ker\_if\_staif.c。

文件ker\_if\_wan.c主要是接口管理注册和解注册函数，如下图所示：

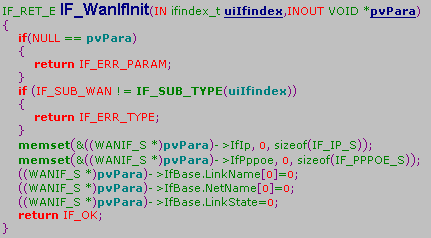


上面两个函数被文件root/ifm/ifm/ker\_if\_main.c中的以下两个函数调用：

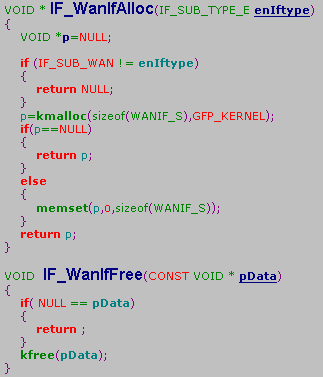


文件root/ifm/ifm/ker\_if\_wanif.c主要是root/ifm/ifm/ker\_if\_wan.c中几个函数的具体定义，

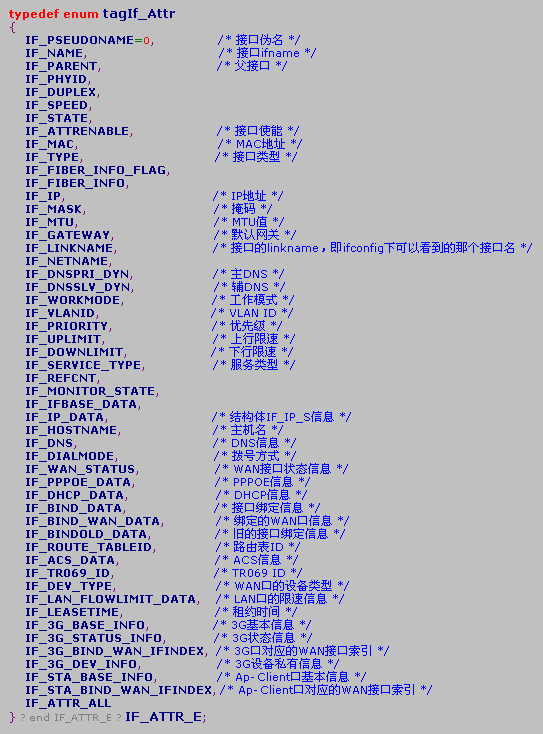
函数IF\_WanIfInit主要是接口数据的初始化：



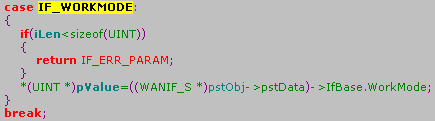
函数IF\_WanIfAlloc和IF\_WanIfFree的作用是接口全局变量的动态分配和释放：



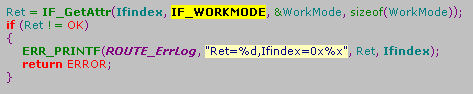
在介绍函数IF\_WanIfGetAttr和IF\_WanIfSetAttr之前，先介绍一下一个很重要的宏，如下图所示：



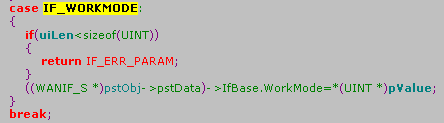
函数IF\_WanIfGetAttr是接口管理的获取接口，只有定义了这个接口，才可以用函数IF\_GetAttr获取上面描述的接口信息，比如我需要获取某个接口的工作模式，则需要在函数IF\_WanIfGetAttr中添加以下语句：



然后我们便可以通过以下语句获取某个接口的工作模式（Ifindex表示接口索引）：



函数IF\_WanIfSetAttr是接口管理的设置函数，只有定义了这个接口，才可以用函数IF\_SetAttr设置上面描述的接口信息，比如我需要设置某个接口的工作模式，则需要在函数IF\_WanIfGetAttr中添加以下语句：



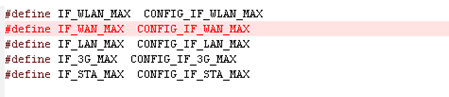
然后我们便可以通过以下语句设置某个接口的工作模式（Ifindex表示接口索引）：



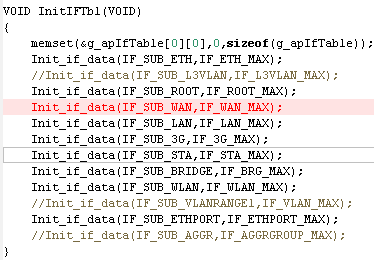
对于其他接口管理变量，也可以利用上面的方法添加。

## 确定接口个数

在头文件root/platform//include/lw\_if\_pub.h中添加表示WAN口个数的宏，如下所示：



然后在文件root/ifm/ifm/ker\_if\_obj.c的函数InitIFTbl中添加如下语句：

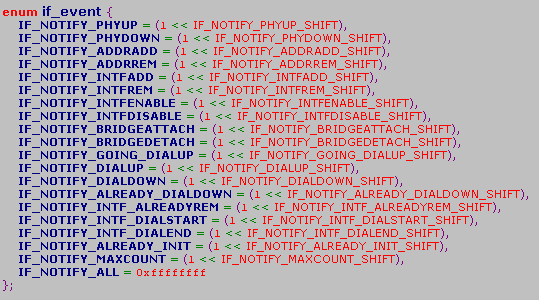


到这里为止，接口管理相关代码差不多已经添加完毕，这个时候运行设备，通过添加打印，就可以看到该接口的相关信息了。

# 接口管理作用

## 接口事件处理

接口是有状态的，比如网线的插拔，链路的连接和断开，地址的增加和删除等都会使得接口的状态发生变化。接口事件处理从本质上讲就是维护这些接口状态。



上图罗列了所有的接口事件类型，目前常用的是以下几个：

IF\_NOTIFY\_PHYUP：接口物理UP事件，接口网线插上时触发；

IF\_NOTIFY\_PHYDOWN：接口物理DOWN事件，接口网线拔掉时触发；

IF\_NOTIFY\_ADDRADD：接口地址增加事件，接口获取到IP地址时触发；

IF\_NOTIFY\_ADDRREM：接口地址删除事件，接口地址被释放时触发；

IF\_NOTIFY\_INTFADD：接口增加事件，接口创建完成时触发；

IF\_NOTIFY\_INTFREM：接口删除事件：接口被删除时触发。

举例：

比如某个接口获取到IP地址了，那么就需要做一些统一的处理，比如下发路由，启动一些业务等，假如没有接口事件处理，那么这些代码都要单独添加，势必造成冗余，如果引入事件处理机制，通过以下这些的方式就可以实现：

上报方代码：

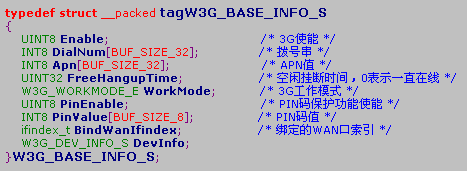
C:\Documents and Settings\Administrator\Application Data\Tencent\Users\844697754\QQ\WinTemp\RichOle\2IYMZRBT%9W@U4(Q[C6KR1L.png

处理方代码：



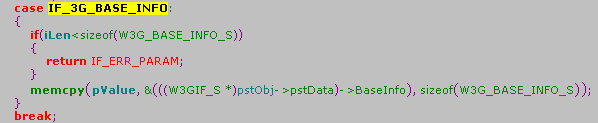
## 保存全局变量

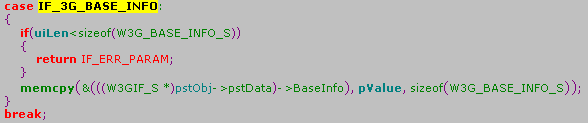
CVNWARE平台中的接口管理模块目前还有一个比较重要的功能，那就是用于保存全局变量，之前保存全局变量的时候，都是单独定义的，比如3G口的一些基本物理属性，之前的做法是单独定义一个结构体，用来保存这些信息，这样做的一个缺点是这些变量只能在master进程中才能访问，其他进程不能直接访问，只能通过IPC通讯后获取。目前的做法是将3G口基本物理属性抽出一个结构体，并把这个结构体作为3G接口对应结构体的一个成员，如下图所述：





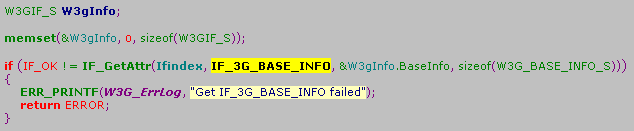
上图中，结构体W3G\_BASE\_INFO\_S是3G基本物理属性结构体，结构体W3GIF\_S是3G接口对应的结构体，几乎所有跟3G口有关的信息都包含在这里了（三层属性除外），将结构体结构体W3G\_BASE\_INFO\_S作为结构体W3GIF\_S的一个成员，就可以通过接口管理属性获取函数获取这些信息了，并且这种方式有一个最大优点，那就是可以跨进程获取和设置，因为这些变量是保存在共享内存里的，当然，前提是必须在文件root/ifm/ifm/ker\_if\_3gif.c文件中添加如下语句：





此后，我们便可以在代码中通过如下语句随时获取和设置3G口的基本物理属性了：

获取3G口的基本物理属性：



设置3G口的基本物理属性：

C:\Documents and Settings\Administrator\Application Data\Tencent\Users\844697754\QQ\WinTemp\RichOle\L3}9J{9KDP4W7JRKG%8{HM5.png

# 接口管理API

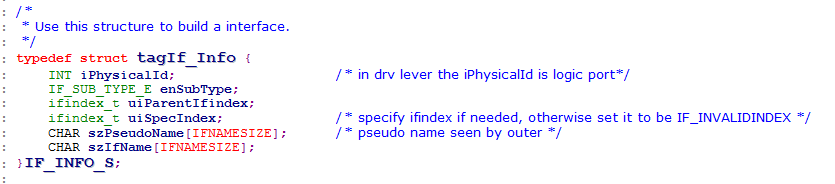
## 索引获取函数

1. IF\_GetByIFName：根据ifname获取接口索引；
2. IF\_GetByPseudoName：根据接口伪名获取索引；
3. ROUTE\_GetIfindexByLinkName：根据linkname获取接口索引；
4. ROUTE\_GetBrIfindexByWanIfindex：根据WAN口索引获取桥口索引。

## 接口创建函数

IF\_RET\_E IF\_CreatInterface( IN IF\_INFO\_S \* pstInfo, OUT ifindex\_t \* puiIfIndex );

入参IF\_INFO\_S 定义如下



enSubType 指定创建接口类型，具体定义在lw\_if\_pub.h中

uiParentIfindex 为父接口索引，存在父接口的，要填写此参数，例如L3VLAN和ETH接口往往存在父子关系。不存在则设置为IF\_INVALID\_IFINDEX

uiSpecIndex 为指定接口索引，如果此参数不为IF\_INVALID\_IFINDEX，系统则为用户创建的接口分配指定的索引，如果设置IF\_INVALID\_IFINDEX，则系统自动分配接口索引

szPseudoName 为伪接口名，用户可以随意设置，接口创建成功后，用户也可以更改

szIfName 为接口名，此接口名一旦设置后，后面将不能变更，其和ifindex都是此接口的

身份标志

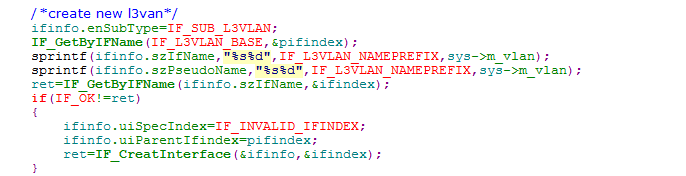
szIfName 的命名有一套规范，其中ETH网口，其名称是netdevice设备名，其他接口命名如下：

L3VLAN IF\_L3VLAN\_NAMEPREFIX+vid  
ETHPORT IF\_ETHPORT\_NAMEPREFIX+logicport

puiIfIndex 为出参，接口创建成功后，系统分配的接口索引

注意：此接口只能在master中使用，一般此接口只用在三层逻辑接口方面，其他物理接口系统会自动创建

举例：



## 接口删除函数

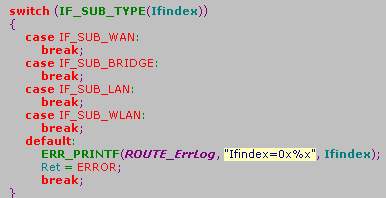
IF\_RET\_E IF\_DeleteInterface( IN ifindex\_t uiIfIndex );

uiIfIndex 为要删除接口的索引

注意： 如果此接口存在子接口，则要先删除字接口，否则无法删除

## 接口类型获取函数

通过宏IF\_SUB\_TYPE获取，如下图所示：



## 接口属性获取函数

IF\_RET\_E IF\_GetAttr( IN ifindex\_t uiIfindex, IN IF\_ATTR\_E enAttrId, INOUT VOID \* pValue ,IN UINT uiLen);

参数说明：

uiIfindex 接口索引

enAttrId 要获取的属性ID

pValue 返回值的BUFFER

uiLen BUFFER 的大小

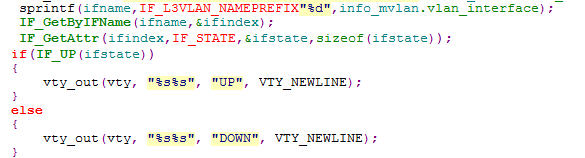
可以获取的属性ID如下：

接口中存在都是实际生效的数据，例如网口可以配置为静态，也可以配置为动态获取

，其中的IP就是实际生效的IP信息

通过此接口可以获取接口状态

例如：



## 接口属性设置函数

IF\_RET\_E IF\_SetAttr( IN ifindex\_t uiIfindex, IN IF\_ATTR\_E iAttrId, IN VOID \* pValue,IN UINT uiLen );

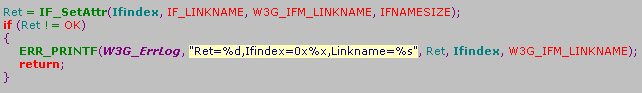
参数说明：

uiIfindex 接口索引

enAttrId 要获取的属性ID

pValue 设置值的BUFFER

uiLen BUFFER 的大小



注意：

此接口只能在MASTER中调用，普通用户一般不要去设置。让master根据系统状态去设置。

## 接口遍历函数

#define IF\_GetFirstIfindex(enSubType,puiIfindex) IF\_GetFirstIfindexByDomain(enSubType,IF\_DOMAIN\_NORMAL,puiIfindex)

获取指定类型接口的首个接口

参数说明：

enSubType 要遍历的接口类型

puiIfindex 返回的接口索引

IF\_RET\_E IF\_GetNextIfindex( IN IF\_SUB\_TYPE\_E enSubType,INOUT ifindex\_t \* puiIfindex );

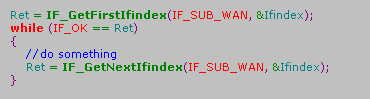
获取指定类型接口的下一个接口，

enSubType 要遍历的接口类型

puiIfindex 入参时，表示当前接口，出参时，表示获取的下一个接口

如果出参 puiIfindex 为IF\_INVALID\_IFINDEX，表示没有接口存在

举例：



## 接口事件相关函数

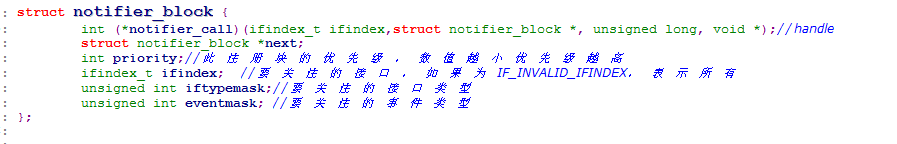
int ifm\_notifier\_register(struct notifier\_block \*nb);

注册关注事件

int ifm\_notifier\_unregister(struct notifier\_block \*nb);

注销关注事件

事件注册块定义如下：



举例：

例如根据接口变化去配置网络地址

