

NS 二十九问之 SeaSon 解答

-From HIT

序言

接触 NS 一年半了，但还是不敢称入门了，NS 实在是博大精深，深奥莫测。面对芸芸高手，斗胆写下 29 问，献给 NS 初学者以供参考，疏漏之处在所难免，其实已经有人指出错误了，但是苦于没有时间纠正，就一直留下了，真是不负责，不像我的作风，但实在太忙了，5555 对不起大家了，请 NSer 大人德，多多见谅，以后会有更新或者后续版本，敬请期待。

对了，是不是还要写点致谢一类的，嘿嘿，谢谢毕业设计阶段石老师对我的指导，不然还不知道 NS 为何物，也不会认识这么多的朋友。谢谢《网络论坛_ NS 仿真软件》让我一步一步成长，谢谢爸爸妈妈，哥哥，女友，liguo 帅小伙，嘿嘿，完了。

最后留下我的联系方式吧：

QQ:67483698（加入时一定要输入“SeaSon 帅哥”，否则不加 HOHO!）

Email: yuhui.wu@163.com（豁出去了，我最常用的，不过不能保证每封信都复，因为一个字忙）

Enjoy it!

2006-2-9 1:08 于家中被窝里^^
SeaSon

目录

一.	怎么样广播数据.....	3
二.	定时循环执行某种操作的函数设计.....	3
三.	发送数据.....	4
四.	IP 数据包的包头.....	4
五.	Hop、经过路径长度的计算方法.....	5
六.	设置 ip 包的源和目的地址.....	5
七.	路由层上传数据.....	5
八.	调用无线节点的路由层中的 <code>command()</code> 里的函数	6
九.	怎样在 C++ 代码里面调用 tcl 函数.....	7
十.	怎样使用 ns 自己的链表.....	8
十一.	转发数据包都需要设置什么?	10
十二.	怎么样改变 nam 中节点的颜色?	10
十三.	谁对谁错? ?	11
十四.	十四.怎样在 c++ 中获得节点地址?	12
十五.	怎样使用定时器?	12
十六.	怎样改变无线网络中 802.11 的能量状态? ...	13
十七.	调试 tr 文件中的 CBK 错误.....	13
十八.	设置物理信道参数.....	14
十九.	CPP_NAMESPACE 为定义的错误解决办法	16
二十.	生成 threshold 工具, 通信参数计算.....	17
二十一.	uptarget_ 和 downtarget_ 怎样逐层的绑定的--	17
二十二.	怎样从 trace 文件中获得从节点 a 发送到 b 的 数据报经历的路径?	18
二十三.	致命的 tab 键?	19
二十四.	Gdb 图形界面调试说明.....	19
二十五.	修改 trace 格式、能量分析.....	23
二十六.	写路由层协议的低级错误, 两次给节点绑定.....	24
二十七.	Error model 让指定时间丢包.....	25
二十八.	跨层设计秘籍-帅哥独家奉送, 嘿嘿.....	25
二十九.	安装问题.....	26

正文

一. 怎么样广播数据

```
Packet *p= allocpkt();
struct hdr_cmh *ch = HDR_CMN(p);
struct hdr_ip *ih = HDR_IP(p);
//初始化
...
...
//关键, 注意还要设置端口, 参见相关文章
ih->daddr() = IP_BROADCAST)
//发送出去, OK 了, 就这么简单
target_->recv(p);
```

二. 定时循环执行某种操作的函数设计

```
/*=====
====
Handler to collect statistics once per second.
其中 MobiHandler 被设为其他类的友元
注意标点符号中英文别搞错了。
-----*/
```

1. 设计 handler 类

```
class MobiHandler : public Handler {
public:
    MobiHandler(RouteCache *C) {
        interval = 1.0;
        cache = C;
    }
    void start() {
        Scheduler::instance().schedule(this, &intr, 1.0);
    }
    void handle(Event *e);
private:
    double interval;
    Event intr;
    RouteCache *cache;
};
```

2. 定义实现 handler () 函数

其中在 routecach 中的定义 handle()的代码如下

```
MobiHandler::handle(Event *intr) {
    //这里是你要周期处理的代码; 放在这里就 OK 了
```

```

        cache->periodic_checkCache();
        Scheduler::instance().schedule(this, intr, interval);
    }
3. 使用计时方法
MobiHandler mh;
mh.start();

```

三. 发送数据

```

    //创建一个数据包
    Packet *p= allocpkt();
    //获得包头相应数据结构的指针
    hdr_cmn::access(p)->size() = size_;
    hdr_rtp* rh = hdr_rtp::access(p);
    //初始化
    rh->flags() = 0;
    rh->seqno() = ++seqno_;
    hdr_cmn::access(p)->timestamp() =
        (u_int32_t)(SAMPLERATE*local_time);
    // add "beginning of talkspurt" labels (tcl/ex/test-rcvr.tcl)
    if (flags && (0 ==strcmp(flags, "NEW_BURST")))
        rh->flags() |= RTP_M;
    p->setdata(data);
    //将数据包发送给下层, OK 了
    target_->recv(p);
//也可以用 Scheduler::instance().schedule(target_, p, t);表示 t 时间后发送

```

四. IP 数据包的包头, 这个需要认真分析了, 不然看代码比较麻烦的说:)

```

struct hdr_ip {
    /* common to IPv{4,6} */
    ns_addr_t  src_;
    ns_addr_t  dst_;
    int        ttl_;

    /* Monarch extn */
    // u_int16_t  sport_;
    // u_int16_t  dport_;

    /* IPv6 */
    int        fid_;    /* flow id */
    int        prio_;

    static int offset_;
    inline static int& offset() { return offset_; }
    inline static hdr_ip* access(const Packet* p) {
        return (hdr_ip*) p->access(offset_);
    }
}

```

```

/* per-field member access functions */
ns_addr_t& src() { return (src_); }
nsaddr_t& saddr() { return (src_.addr_); }
int32_t& sport() { return src_.port_;}

ns_addr_t& dst() { return (dst_); }
nsaddr_t& daddr() { return (dst_.addr_); }
int32_t& dport() { return dst_.port_;}
int& ttl() { return (ttl_); }
/* ipv6 fields */
int& flowid() { return (fid_); }
int& prio() { return (prio_); }
};

```

五. Hop、经过路经长度的计算方法

```

struct hdr_ip *ih = HDR_IP(p);
ih->ttl_ = TTL_START;//ttl 的初始值
ih->ttl_ -= 1//每经过一个节点减少 1
//其实还可以利用它来计算路经长度,如下, 一般人我还不告诉他呢: )
Route_length=TTL_START-ih->ttl

```

六. 设置 ip 包的源和目的地址

因为都是结构体, 所以函数都可以直接访问

```

ih->saddr() = index;
ih->daddr() = ipdst;
ih->sport() = RT_PORT;
ih->dport() = RT_PORT;

```

七. 路由层上传数据, 这里有点问题, 回学校后我再更正, 也欢迎你给我提供你的理解

参见 aodv.cc

1. 在 aodv.h 中声明


```

PortClassifier *dmux_;

```
2. if (ch->ptype() != PT_AODV && ch->direction() == hdr_cmn::UP && ((u_int32_t)ih->daddr() == IP_BROADCAST) || ((u_int32_t)ih->daddr() == here_.addr_)) {


```

dmux_->recv(p,0); //传递给分类器
return;

```

3. 分类器处理数据包

```

void Classifier::recv(Packet* p, Handler*h)
{
    NsObject* node = find(p);//查找节点的是否存在
    if (node == NULL) {
        /*
         * XXX this should be "dropped" somehow. Right now,
         * these events aren't traced.
         */
    }
}

```

```

    Packet::free(p);
    return;
}
node->recv(p,h);//让节点接受

```

4.节点接受后

```

. void NsObject::recv(Packet *p, const char*)
{
    Packet::free(p);
}

```

八. 怎么调用无线节点的路由协议中的 **command()** 里面的函数
这个与其它的有点不同, 所以单独拿来分析, 希望对你有帮助;

例如, 我的路由协议 myrtagent 的 **command()** 里面有 **seta** 方法如下面红色字体所示。则在 **tcl** 脚本中可以用一下方式调用这个方法;

```

[$node_(0) agent 255] seta 10

```

[\$node_(0) agent 255]表示获得 **node_(0)**的 255 端口的代理 (协议)
然后再调用这个代理的 **seta** 函数。

```

Flood::command(int argc, const char*const* argv) {
    Tcl& tcl = Tcl::instance();
    if(argc == 2) {
        if(strncasecmp(argv[1], "id", 2) == 0) {
            cout<<"The ip addr is "<<argv[4]<<endl;
            tcl.resultf("mflood:%d", index_);
            return TCL_OK;
        }
        else if (strcmp(argv[1], "geta") == 0){
            cout<<"haha geta :"<<a<<endl;
            return TCL_OK;
        }
        else if (strcmp(argv[1], "uptarget") == 0) {
            if (uptarget_ != 0)
                tcl.result(uptarget_->name());
            return (TCL_OK);
        }
    } else if(argc == 3) {
        if(strcmp(argv[1], "index_") == 0) {
            index_ = atoi(argv[2]);
            return TCL_OK;
        } else if(strcmp(argv[1], "log-target") == 0 || strcmp(argv[1], "tracetarget")
== 0) {
            logtarget = (Trace*) TclObject::lookup(argv[2]);
            if(logtarget == 0) return TCL_ERROR;
            return TCL_OK;
        }
        else if (strcmp(argv[1], "seta") == 0){
            a=atoi(argv[2]);
            return TCL_OK;
        }
        else if (strcmp(argv[1], "uptarget") == 0) {

```



```

$ns_ connect $udp_(0) $null_(0)

set cbr_(0) [new Application/Traffic/CBR]
$cbr_(0) set packetSize_ 512
$cbr_(0) set interval_ 4.0
$cbr_(0) set random_ 1
$cbr_(0) set maxpkts_ 10000
$cbr_(0) attach-agent $udp_(0)
$ns_ at $starttm "$cbr_(0) start"
$ns_ at [expr $starttm+30.0 ] "$cbr_(0) stop"

}

```

十. 怎样使用 ns 自己的链表

1. 头文件 `#include<lib/bsd-list.h>`

2. 初始化

1). 在节点中添加，包括了指向前驱和后继的节点。

```

#define LIST_ENTRY(type)
struct {
    type *le_next; /* next element */
    type **le_prev; /* address of previous next element */
}

```

2). 定义一个链表，链表的类型为 type, 表头为 lh_first

```

#define LIST_HEAD(name, type)
struct name {
    type *lh_first; /* first element */
}

```

3) 初始化链表

```

#define LIST_INIT(head) {
    (head)->lh_first = NULL;
}

```

4) 插入节点

```

#define LIST_INSERT_AFTER(listelm, elm, field) {
    if (((elm)->field.le_next = (listelm)->field.le_next) != NULL)
        (listelm)->field.le_next->field.le_prev =
            &(elm)->field.le_next;
    (listelm)->field.le_next = (elm);
    (elm)->field.le_prev = &(listelm)->field.le_next;
}

```

```

#define LIST_INSERT_BEFORE(listelm, elm, field) {
    (elm)->field.le_prev = (listelm)->field.le_prev;
    (elm)->field.le_next = (listelm);
    *(listelm)->field.le_prev = (elm);
    (listelm)->field.le_prev = &(elm)->field.le_next;
}

```

```

#define LIST_INSERT_HEAD(head, elm, field) {
    if (((elm)->field.le_next = (head)->lh_first) != NULL)

```

```

    (head)->lh_first->field.le_prev = &(elm)->field.le_next;\
    (head)->lh_first = (elm);
    (elm)->field.le_prev = &(head)->lh_first;
}

```

5) 删除节点

```

#define LIST_REMOVE(elm, field) {
    if ((elm)->field.le_next != NULL)
        (elm)->field.le_next->field.le_prev =
            (elm)->field.le_prev;
    *(elm)->field.le_prev = (elm)->field.le_next;
}

```

6) 一个例子

(1) 定义节点类型

```

class MFlood_RTEntry {
    friend class MFlood_RTable;
    friend class MFlood;

public:
    MFlood_RTEntry();
    MFlood_RTEntry(nsaddr_t src,u_int32_t seq);
    bool    isNewSeq(u_int32_t seq);    // old -> false, new->true
    void    addSeq(u_int32_t seq);    // add a seqno to seqno array(rt_seqnos)

```

protected:

```

    LIST_ENTRY(MFlood_RTEntry) rt_link;

    nsaddr_t src_;
    // u_int32_t seq_;

    u_int32_t    rt_seqnos[REM_SEQ_COUNT];    //seqno array
    u_int32_t    max_seqno; //max seqno
    u_int32_t    min_seqno; //max seqno
    u_int16_t    seq_it; // seqno's iterator
};

```

(2) 建立一个链表节点类型为 MFlood_RTEntry 的链表 rthead

```
LIST_HEAD(, MFlood_RTEntry) rthead;
```

(3) 初始化链表 rthead 为 NULL

```
LIST_INIT(&rthead)
```

(4) 怎样使用

```

MFlood_RTEntry*
MFlood_Rtable::rt_lookup(nsaddr_t id) {
    MFlood_RTEntry *rt = rthead.lh_first; //获取链表表头
    for(; rt; rt = rt->rt_link.le_next) {
        if(rt->src_ == id)
            break;
    }
    return rt;
}

```

(5) 删除节点

```

void
MFlood_RTable::rt_delete(nsaddr_t id) {
    MFlood_RTEntry *rt = rt_lookup(id);
    if(rt) {
        LIST_REMOVE(rt, rt_link);
        delete rt;
    }
}

```

(6) 插入节点

```

rt = new MFlood_RTEntry(ih->saddr(), fh->seq_);
LIST_INSERT_HEAD(&rtable_rthead,rt,rt_link);

```

十一. 转发数据包都需要设置什么？

我们需要改变一下参数，不然你会死得很惨的，嘿嘿

```

ch->addr_type()=NS_AF_INET;
ch->direction()=hdr_cmn::DOWN;
send(p);

```

十二. 怎么样改变 nam 中节点的颜色？

Lars Wischhof give the following message and It is work.:

It works if you do it in the following way:

```

# mark node 12
$node_(12) color black #在创建节点的时候添加这条语句，增加 color 属性
                        #否则气死你也不会变色，呵呵
$ns_ at 30.0 "$node_(12) color red"

```

This changes the color of node 12 at time 30.0 to red. The first of the two commands is needed so that the node has a color property -- it does not really have an effect in nam.

节点的颜色。。

```

#Define color index
$ns color 0 red
$ns color 1 yellow
$ns color 2 blue
$ns color 3 green
$ns color 4 black
$ns color 5 chocolate
$ns color 6 brown
$ns color 7 tan
$ns color 8 gold
$ns color cyan

```

形状

```
circle
box
hexagon
```

十三. 谁对谁错??

```
set hda 10
//错误代码?? //
if {$hda==0} {
    $node_color red
}
elseif {$hda==1} {
    $node_color blue
} elseif {$hda==2} {
    $node_color yellow
}
elseif {$hda==3} {
    $node_color brown
}
elseif {$hda==4} {
    $node_color tan
}
else{
    $node_color gold
}
//////////正确代码
if {$hda == 0} {
    $node_color red
} elseif {$hda == 1} {
    $node_color blue
} elseif {$hda == 2} {
    $node_color yellow
} elseif {$hda == 3} {
    $node_color brown
} elseif {$hda == 4} {
    $node_color tan
} else {
    $node_color gold
}
}
```

结论: 关于 TCL 的看不见的错误, 很多都是在句末多了 **tab** 键, 这个问题是看不出来, 需要全部选择, 根据长度判定, 你是不是头大了, 告诉你, 我也是。

十四.怎样在 c++中获得节点地址?

```
suppose that in Tcl you set the node's id to 0
(NOTE: I'm not sure that the 'id' attribute exists!
It could be 'id_' or whatever: this is just to
make you understand the mechanism...)
```

```
$node set id 0
```

then in C++ you can access the id with:

```
Tcl& tcl = Tcl::instance();
tcl.evalc("$node set id");
const char* addr = tcl.result();
int node_id = atoi(addr);
The node's id is now stored in 'node_id'.
```

十五.怎样使用定时器?

- 1, 定义一个定时器类, 其中你处理超时的代码写在 `expire()`函数里面就行了。

```
class Mfloodtimer : public TimerHandler {
public:
    Mfloodtimer(MFlood* t) : TimerHandler(), t_(t) {}
    inline virtual void expire(Event*e);
protected:
    MFlood * t_;//在这个类里面使用定时器
};
```

2. 超时执行的代码

```
void Mfloodtimer::expire(Event *e){
    //如果你要重新启动定时器, 这里面最好是调用 t_里面的函数,可以在里面
    //调用 timer_.resched(10);重新启动定时器。
    //在这里只供演示显示超时了

    cout<<"!!!!At "<<NOW<<" mflood expried:(\n"<<endl;

}
}
```

3. 在你的协议函数里面怎样使用。

```
Mfloodtimer *timer_=new Mfloodtimer(this); //this 是你的 Agent
timer_->sched(10);
```

```
hdrip->src_.addr_ >> Address::instance().NodeShift_[1],? ?
```

十六.怎样改变无线网络中 802.11 的能量状态?

```
# Power vs. Range
```

```
# Assume AT&T's Wavelan PCMCIA card -- Chalermek
```

```
# Pt_ = 8.5872e-4; // For 40m transmission range.
```

```
# Pt_ = 7.214e-3; // For 100m transmission range.
```

```
# Pt_ = 0.2818; // For 250m transmission range.
```

```
# Pt_ = pow(10, 2.45) * 1e-3; // 24.5 dbm, ~ 281.8mw
```

```
Phy/WirelessPhy set Pt_ 8.5872e-4
```

十七.调试 tr 文件中的 CBK 错误

这是我在我当时写的协议中遇到的一个棘手的问题，希望能起到抛砖引玉的作用。

```
D 122.040067803 _4_ RTR CBK 257 Mrpqos 20 [13a 21 4 800] ----- [4:0 0:0 30 33]
```

```
D 122.073627803 _4_ RTR CBK 258 Mrpqos 20 [13a 20 4 800] ----- [4:0 27:0 30 32]
```

1. 追根求源，问题所在分析

```
AODV::rt_ll_failed(Packet *p) {  
    struct hdr_cmn *ch = HDR_CMN(p);  
    struct hdr_ip *ih = HDR_IP(p);  
    aodv_rt_entry *rt;  
    nsaddr_t broken_nbr = ch->next_hop_;
```

```
#ifndef AODV_LINK_LAYER_DETECTION  
    printf("%%%drop type1\n");  
    drop(p, DROP_RTR_MAC_CALLBACK);  
#else
```

```
    /*  
     * Non-data packets and Broadcast Packets can be dropped.  
     */
```

//因为我的数据包不含有实际数据所以在下面 DATA_PACKET() 函数的判断中被丢弃了

//所以关键就是修改者个函数，具体修改参见 2.修改

```
DATA_PACKET(ch->ptype())
```

```
    if(! DATA_PACKET(ch->ptype()) ||  
        (u_int32_t) ih->daddr() == IP_BROADCAST) {  
        printf("%%%drop type2\n %d",ch->ptype());//测试丢包具体原因所在。  
        drop(p, DROP_RTR_MAC_CALLBACK);  
        return;  
    }  
    log_link_broke(p);  
    if((rt = rtable.rt_lookup(ih->daddr())) == 0) {  
        printf("%%%drop type3\n");  
        drop(p, DROP_RTR_MAC_CALLBACK);  
        return;  
    }  
    log_link_del(ch->next_hop_);
```

```
#ifdef AODV_LOCAL_REPAIR
```

```
    /* if the broken link is closer to the dest than source,  
       attempt a local repair. Otherwise, bring down the route. */
```

```

if (ch->num_forwards() > rt->rt_hops) {
    local_rt_repair(rt, p); // local repair
    // retrieve all the packets in the ifq using this link,
    // queue the packets for which local repair is done,
    return;
}
else
#endif // LOCAL REPAIR

{
    printf("%%%drop type4\n");
    drop(p, DROP_RTR_MAC_CALLBACK);
    // Do the same thing for other packets in the interface queue using the
    // broken link -Mahesh
while((p = ifqueue->filter(broken_nbr))) {
    printf("%%%drop type5\n");
    drop(p, DROP_RTR_MAC_CALLBACK);
}
    nb_delete(broken_nbr);
}

#endif // LINK LAYER DETECTION
}

```

2.修改 DATA_PACKET(ch->ptype())

```

#define DATA_PACKET(type) ( (type) == PT_TCP || \
                             (type) == PT_TELNET || \
                             (type) == PT_CBR || \
                             (type) == PT_AUDIO || \
                             (type) == PT_VIDEO || \
                             (type) == PT_ACK || \
                             (type) == PT_SCTP || \
                             (type) == PT_SCTP_APP1 || \
                             //解决问题的关键所在!!!
                             (type) == PT_MRPQOS \
                             )

```

十八.设置物理信道参数 mail-list\1999-August.txt

```

# set up the antennas to be centered in the node and 1.5 meters above it
Antenna/OmniAntenna set X_ 0
Antenna/OmniAntenna set Y_ 0
Antenna/OmniAntenna set Z_ 1.5
Antenna/OmniAntenna set Gt_ 1.0
Antenna/OmniAntenna set Gr_ 1.0

# Initialize the SharedMedia interface with parameters to make

```

```
# it work like the 914MHz Lucent WaveLAN DSSS radio interface
Phy/WirelessPhy set CPTthresh_ 10.0
Phy/WirelessPhy set CSTthresh_ 1.559e-11
Phy/WirelessPhy set RXThresh_ 3.652e-10
Phy/WirelessPhy set Rb_ 2*1e6
Phy/WirelessPhy set Pt_ 0.2818
Phy/WirelessPhy set freq_ 914e+6
Phy/WirelessPhy set L_ 1.0
```

```
### set opt(chan)                Channel/WirelessChannel
这些设置要在 set chan          [new $opt(chan)]之前设置
```

//这些代码是我要将网络节点分类所用的，没什么用处，但还要留着，嘿嘿

```
Agent/Mrpqos instproc nodecolor {hda} {
    global ns_ tracefd  namtrace
    $self instvar node_
    if {$hda == 0} {
        $node_ color red
    } elseif {$hda == 1} {
        $node_ color blue
    } elseif {$hda == 2} {
        $node_ color yellow
    } elseif {$hda == 3} {
        $node_ color brown
    } elseif {$hda == 4} {
        $node_ color tan
    } else {
        $node_ color green
    }
    $node_ label $hda
    # $node_ label-color green
}
```

```
Agent/Mrpqos instproc nodecolor {hda} {
    global ns_ tracefd  namtrace
    $self instvar node_
    if {$hda == 0} {
        $node_ add-mark m2 purple circle
        $node_ color red
    } elseif {$hda == 1} {
        $node_ add-mark m2 purple circle
        $node_ color blue
    } elseif {$hda == 2} {
        $node_ add-mark m2 purple circle
        $node_ color yellow
    } elseif {$hda == 3} {
        $node_ add-mark m2 purple circle
        $node_ color brown
    }
}
```

```

} elseif {$hda == 4} {
  $node_ add-mark m2 purple circle
  $node_ color tan
} elseif {$hda == 5} {
  $node_ add-mark m2 purple circle
  $node_ color cyan
} elseif {$hda == 6} {
  $node_ add-mark m2 purple hexagon
  $node_ color red
} elseif {$hda == 7} {
  $node_ add-mark m2 purple hexagon
  $node_ color blue
} elseif {$hda == 8} {
  $node_ add-mark m2 purple hexagon
  $node_ color yellow
} elseif {$hda == 9} {
  $node_ add-mark m2 purple hexagon
  $node_ color brown
} elseif {$hda == 10} {
  $node_ add-mark m2 purple hexagon
  $node_ color tan
} elseif {$hda == 11} {
  $node_ add-mark m2 purple hexagon
  $node_ color cyan
} elseif {$hda == 12} {
  $node_ add-mark m2 purple circle
  $node_ color green
} elseif {$hda == 13} {
  $node_ add-mark m2 purple hexagon
  $node_ color green
} elseif {$hda == 14} {
  $node_ add-mark m2 purple square
  $node_ color red
} elseif {$hda == 15} {
  $node_ add-mark m2 purple square
  $node_ color red
} else {
  $node_ color black
}
$node_ label $hda
# $node_ label-color green
}

```

十九.CPP_NAMESPACE 为定义的错误解决办法

在 autoconfig.h 中添加

```
#define CPP_NAMESPACE std
```

重新编译就可以了

二十.生成 **threshold** 工具，通信参数计算。

Go to indep-utils/propagation directory in ns installation folder. Compile the file threshold.cc [In my case, "g++ -lm -o threshold threshold.cc"]

进入 indep-utils/propagation 目录,编译 threshold.cc 文件(命令:g++ -lm -o threshold threshold.cc) , 然后根据下面的说明使用即可。

参数说明:

\$./threshold

USAGE: find receiving threshold for certain communication range (distance)

SYNOPSIS: threshold -m <propagation-model> [other-options] distance

<propagation-model>: FreeSpace, TwoRayGround or Shadowing

[other-options]: set parameters other than default values:

Common parameters:

-Pt <transmit-power>

-fr <frequency>

-Gt <transmit-antenna-gain>

-Gr <receive-antenna-gain>

-L <system-loss>

For two-ray ground model:

-ht <transmit-antenna-height>

-hr <receive-antenna-height>

For shadowing model:

-pl <path-loss-exponent>

-std <shadowing-deviation>

-d0 <reference-distance>

-r <receiving-rate>

二十一.**uptarget_**和 **downtarget_**怎样一层一层的绑定的。

参见 ns-mobilenode.tcl 下列代码，会对你有所启发的

```
-----  
if {$simepflag == "ON" } {  
    $simep recvtarget [$self entry]  
    $simep sendtarget $ll  
    $ll up-target $simep  
    } else {  
    $ll up-target [$self entry]  
    }  
#
```

```

# Interface Queue
#
$ifq target $mac
$ifq set limit_ $qlen
if { $imepflag != "" } {
    set drpT [$self mobility-trace Drop "IFQ"]
} else {
    set drpT [cmu-trace Drop "IFQ" $self]
}
$ifq drop-target $drpT
if { $namfp != "" } {
    $drpT namattach $namfp
}
#
# Mac Layer
#
$mac netif $netif
$mac up-target $ll

if { $outerr == "" && $fec == "" } {
    $mac down-target $netif
} elseif { $outerr != "" && $fec == "" } {
    $mac down-target $outerr
    $outerr target $netif
} elseif { $outerr == "" && $fec != "" } {
    $mac down-target $fec
    $fec down-target $netif
} else {
    $mac down-target $fec
    $fec down-target $outerr
    $err target $netif
}
}

```

二十二.怎样从 trace 文件中获得从节点 a 发送到 b 的数据报经历的路径?

你也可以利用这种方法查找你数据报载什么地方丢了，方便你解决这些问题。

如下面所示：

```
$ gawk '$14=="[a:0" && $15~/b:0/ {print}' dasele.tr
```

其中\$14，\$15 代表第 14 列和 15 列（以空格分开），如：

```
s 69.530259667 _115_ AGT --- 326 Mrpqos 0 [0 0 0 0] ----- [115:0 23:0 32 0]
s 69.530259667 _115_ RTR --- 326 Mrpqos 0 [0 0 0 0] ----- [115:0 23:0 32 0]
```

举例：

```
Candy@CandySeaSon /cygdrive/d/my_programs/ns2/test/mrpqos
```

```
$ gawk '$14=="[115:0" && $15~/23:0/ {print}' dasele.tr
```

```
s 69.530259667 _115_ AGT --- 326 Mrpqos 0 [0 0 0 0] ----- [115:0 23:0 32 0]
```

```
r 69.530259667 _115_ RTR --- 326 Mrpqos 0 [0 0 0 0] ----- [115:0 23:0 32 0]
```

```
s 69.530259667 _115_ RTR --- 326 Mrpqos 20 [0 0 0 0] ----- [115:0 23:0 30 44]
r 69.535174494 _44_ RTR --- 326 Mrpqos 20 [13a 2c 73 800] ----- [115:0 23:0 30 44]
f 69.535174494 _44_ RTR --- 326 Mrpqos 20 [13a 2c 73 800] ----- [115:0 23:0 29 90]
r 69.537091728 _90_ RTR --- 326 Mrpqos 20 [13a 5a 2c 800] ----- [115:0 23:0 29 90]
f 69.537091728 _90_ RTR --- 326 Mrpqos 20 [13a 5a 2c 800] ----- [115:0 23:0 28 23]
r 69.538990075 _23_ AGT --- 326 Mrpqos 20 [13a 17 5a 800] ----- [115:0 23:0 28 23]
```

从上面的代码可知道从节点 115 到节点 23 的路径是：115-> 44-> 90-> 23。

二十三.致命的 tab 键?

在 tcl 中对 tab 键比较敏感，如 ice 曾经遇到的问题

```
-phyType $val(netif) \           #这里面多了几个 tab
-topoInstance $topo \
```

因为那几个 tab 键导致运行的时候提示下列错误。

```
$ ns mesp.tcl
num_nodes is set 2
invalid command name "-topoInstance"
    while executing
"-topoInstance $topo \
    -agentTrace ON \
    -routerTrace ON \
    -macTrace OFF \
    -movementTrace OFF \
    -channel $chan_1_           "
(file "mesp.tcl" line 78)
```

所以这种细节的问题大家一定要多多注意。因为他是根本看不出来的。

二十四.Gdb 图形界面调试说明

1. 安装再 cygwin 下面安装就行了,注意我测试的 gdb-20020411-1 和 gdb-20010428-3 都是图形界面,而版本 gdb-20041228-3 我发现是命令行的,所以

推荐前两个版本，毕竟图形界面方便很多，如果你热衷于新的版本，那看到这里你就可以停止了：）下面以 ns-allinone-2.27 为例说明

2. 修改 Makefile(没有任何后缀的)将里面

CCOPT = #如果是 ns-allinone-2.28,这里是 **CCOPT = -O2**

STATIC=

LDFLAGS = \$(STATIC)

LDOUT= -o \$(BLANK)

改变为:

CCOPT = -g #如果是 ns-allinone-2.28,这里修改为 **CCOPT = -O2 -g**

STATIC=

LDFLAGS = \$(STATIC)

LDOUT= -o \$(BLANK)

修改的东西是红色字体所示

3. 然后重新编译

Make clean

Make depend

Make

4.调试过程如下

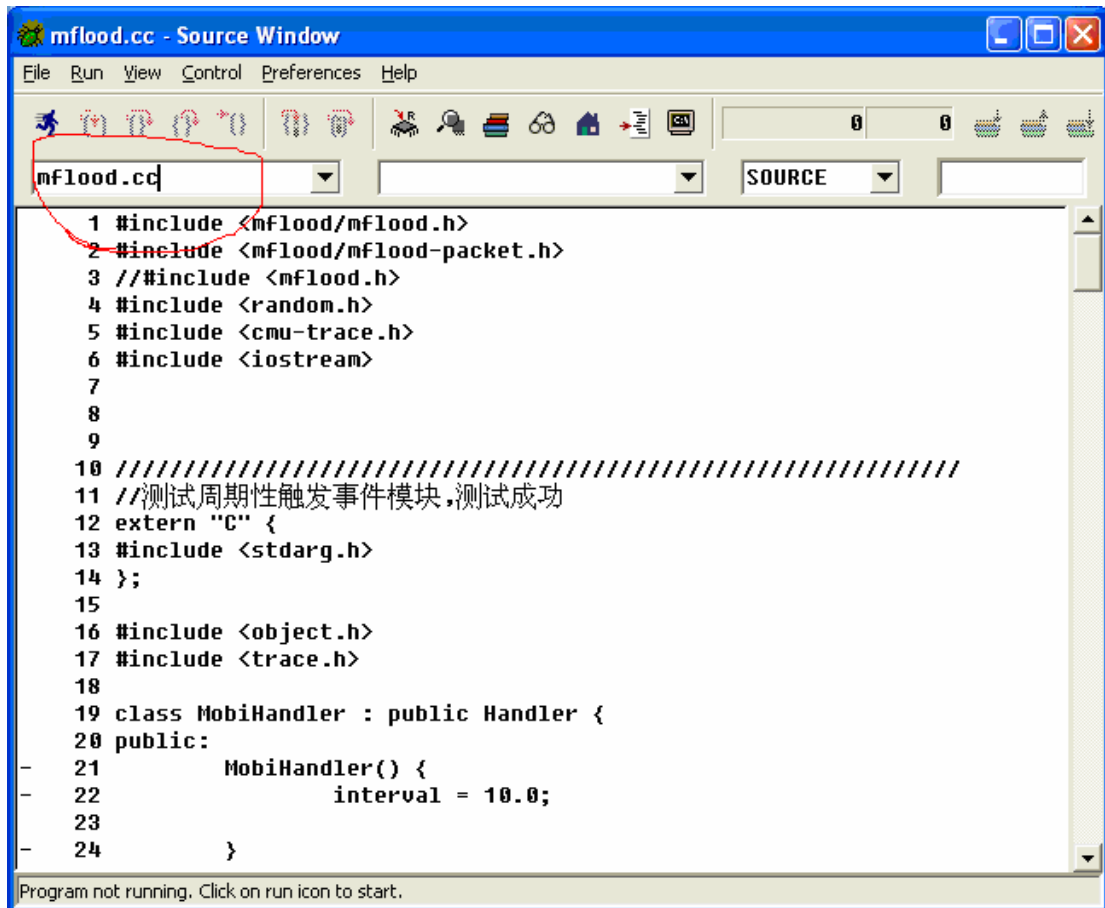
进入工作目录：推荐 ns-2.27

操作：打开 xwindows 之后，

a.输入 **cd ns-allinone-2.27/ns-2.27** (必须)

b.输入**gdb ns**进入gdb工作界面，如果不是图形界面的版本可能就直接在xwindows进入Gdb调试过程，这样的话具体命令参见《NS仿真软件》精华区，不要问我！！！！！！！！

如果按照我的要求安装的图形界面 gdb，出现如下图形界面，如下所示：



c.在左上角红色标记的地方输入你要调适的程序。如:mflood.cc

设置断点，注意只有最左边有“-”符号（如下图中 1 所示）的可以设置断点，将鼠标放在行号上就会出现变成一个圆形的符号，单击即设了一个断点，原来的“-”地方变成了一个红色的方框，如下图所示中 2 所示。然后输入 Ctrl+N 进入命令行窗口，如下图所示，

在命令行窗口进入你的 tcl 脚本所在的目录：

我的在 ns-2.27/mytcl

因为当前的工作目录在 ns-2.27，所以我直接输入 cd mytcl 就行了
输入命令 r mytcl.tcl 就进入调试，具体调试跟 vc 差不多，呵呵，不用我罗嗦了，一些常用的如下图所示，其他的你看英文就知道意思了。祝好运

The image shows a debugger interface with two windows: "mflood.cc - Source Window" and "Console Window".

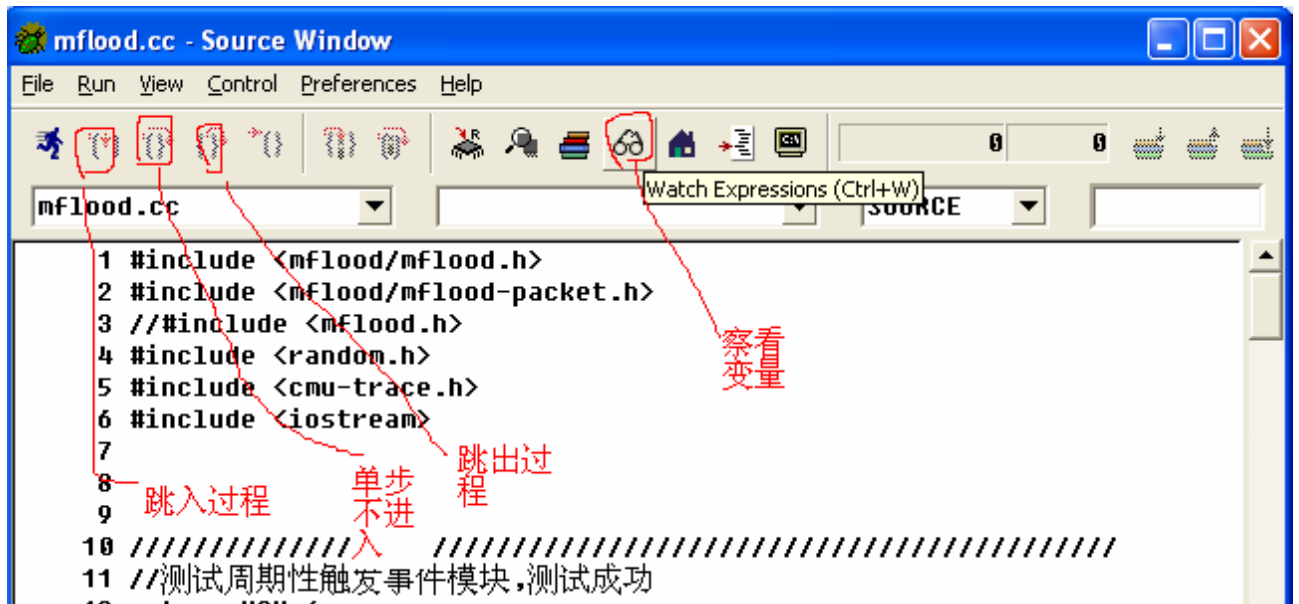
mflood.cc - Source Window:

- Menu: File, Run, View, Control, Preferences, Help
- Toolbar: Contains various icons for navigation and execution.
- File Name: mflood.cc (with handwritten red text "调试" next to it)
- View: SOURCE
- Code:

```
1 #include <mflood/mflood.h>
2 #include <mflood/mflood-packet.h>
3 // #include <mflood.h>
4 #include <random.h>
5 #include <cmu-trace.h>
6 #include <iostream>
7
8
9
10 //////////////////////////////////////
11 //测试周期性触发事件模块,测试成功
12 extern "C" {
13 #include <stdarg.h>
14 };
15
16 #include <object.h>
17 #include <trace.h>
18
19 class MobiHandler : public Handler {
20 public:
21     MobiHandler() {
22         interval = 10.0;
23     }
24 }
```
- Breakpoint: A red square breakpoint is set at line 22. A tooltip shows: "breakpoint 1 at mflood.cc:22 (0x61c1b4) ENA breakpoint donttouch threads=all cond=None".
- Status Bar: breakpoint 1 at mflood.cc:22 (0x61c1b4)

Console Window:

- Header: Console Window
- Content: (gdb) |



二十五.修改 trace 格式、能量分析

Energy-model.cc

```
void EnergyModel::DecrIdleEnergy(double idletime, double P_idle)
{
    double dEng = P_idle * idletime;
    if (energy_ <= dEng)
        energy_ = 0.0;
    else
        energy_ = energy_ - dEng;
    if (energy_ <= 0.0)
        God::instance()->ComputeRoute();
}
```

Trace 中的 energy 表示节电剩余能量。Ns-2.27\trace\cmu-trace.cc

```
if (thisnode) {
    if (thisnode->energy_model()) {
        sprintf(pt_->buffer() + offset,
            "[energy %f]",
            thisnode->energy_model()->energy());
    }
}
```

```
#if COMMENT_ONLY
```

```
-----
|           |
| Upper Layers |
|           |
-----
|           |
```


二十七、Error model 让指定时间丢包

```
# create a loss_module and set its packet error rate to 1 percent
set loss_module [new ErrorModel]
$loss_module set rate_ 0.01
# optional: set the unit and random variable
$loss_module unit pkt ;# error unit: packets (the default)
$loss_module ranvar [new RandomVariable/Uniform]
# set target for dropped packets
$loss_module drop-target [new Agent/Null]
```

在 `errormodel.h` 上面增加这一行，主要为了获得时间，也可以直接调用括号中的东西。

```
#define NOW_ (Scheduler::instance().clock())
```

把 `errormodel.cc` 中下面的函数修改成这样

```
int ErrorModel::CorruptPkt(Packet*)
{
    // if no random var is specified, assume uniform random variable
    //double u = ranvar_ ? ranvar_->value() : Random::uniform();
    //return (u < rate_);
    //因时间感觉不一定正好那个时间收到一个包，所以最好用一个范围看看，呵呵
    if(NOW_==yourtime1 || NOW_==yourtime2 ... ..)
        return 1;//不行的话改称 0 赫赫
}
```

重新编译，然后

调用方法

```
# create a loss_module and set its packet error rate to 1 percent
set loss_module [new ErrorModel]
$loss_module set rate_ 0.01
# optional: set the unit and random variable
$loss_module unit pkt ;# error unit: packets (the default)
$loss_module ranvar [new RandomVariable/Uniform]
# set target for dropped packets
$loss_module drop-target [new Agent/Null]
```

因为我打开了路由层的 `trace`，所以在 `cmu-trace.cc`，定义一个 `CMUTrace` 的共有变量 `mytarget`，用来指向链路层，初始化在他的 `command`

二十八、跨层设计秘籍-帅哥独家奉送，嘿嘿

怎样在任意层访问下层的信息（包括 `netif`，`mac`，`ifq`，`ll` 等）

以 aodv 中访问 mac 为例,

1. 在 aodv.h 中增加头文件

```
#include "mac/mac-802_11.h"
```

在 AODV 类里面声明

```
Mac802_11 * mymac;
```

2. 修改 command()函数, 增加以下代码

```
int
AODV::command(int argc, const char*const* argv) {
    . . .
    . . .
    else if(argc == 3) {
        if(strcmp(argv[1], "index") == 0) {
            index = atoi(argv[2]);
            return TCL_OK;
        }//add by season
        else if (strcmp(argv[1], "set-mac") == 0)
        {
            mymac = (Mac802_11 *) TclObject::lookup(argv[2]);
            if (mymac == 0)
            {
                fprintf(stderr, "MESPAgent: %s lookup %s failed.\n", argv[1], argv[2]);
                return TCL_ERROR;
            }
            else
            {
                //test
                printf("Get Node mac bss_id:%d \n", mymac->bss_id());
                // fprintf(stderr, "Get Node address .\n", nodeID);

                return TCL_OK;
            }
        }
        . . .
        . . .
    }
}
```

3. 在 tcl 脚本中初始化

```
set rt($i) [$node_($i) agent 255] # 获得路由层协议
```

```
$rt($i) set-mac [$node_($i) set mac_(0)] #初始化 mac 对象
```

4. 然后就可以在 AODV 中通过 mymac 对象访问 mac 的信息

二十九、安装问题

提示: 如果安装过程中出现任何 error, 统统都是安装失败的标志, 所以出现种种不良症状均是正常现象, 比如 ns 不能启动了, nam 无法显示了等等。这时候你就需要求助 Google 或者论坛了, 下面是两个问题

1、winxp 下装 cygwin+ns2.27allinone 错误:

```
make:***[install-libraries]Error 254
```

```
tcl8.4.5 installation failed.
```

```
tcl is not part of the ns project. please see www.scriptics.com to see if they have a fix for your platform
```

p.s. 已经打了 ns2.27allinone 在 cygwin 的补丁。完全按照 <http://140.116.72.80/~smallko/ns2/setup.htm> 的步骤做的, cygwin 的部件该装的都装了。

请各位不吝指教。

solution: 安装 cygwin 的时候全部选上 x11 所有的包

2、好像是 tcltk 什么出现 C++语法错误

这时 gcc 版本所限制, 换低版本 3.1*以下, 现在网上的好像 Cygwin 好像没有了, Google 之, 或者我的公用信箱里面有。