



دانشکده مهندسی کامپیوتر

گزارش کار پروژه NS2

(پایاده سازی حمله تصرف در شبکه‌های MANET)

نام دانشجویان:

حسین رحمتی زاده ذاقلی ۸۸۵۲۱۱۳۹

عماد آقاجانی ۸۸۵۲۱۳۴۴

استاد راهنما:

دکتر زینب موحدی

تیر ماه ۱۳۹۳

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

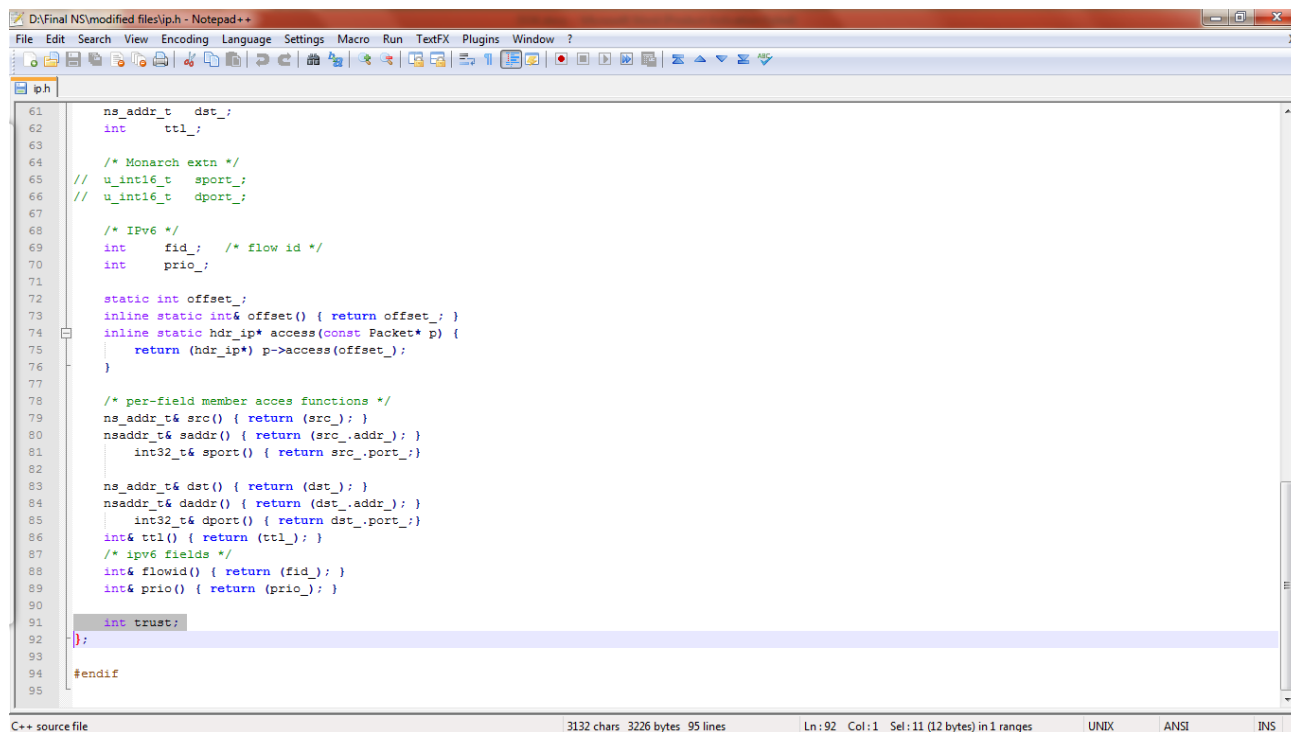
مقدمه:

گزارش کار:

در این پروژه در یک شبکه‌ی MANET گروهی از گره‌ها را به عنوان گره‌های متخاصم و گروهی دیگر را به عنوان گره‌های معمولی می‌نامیم که فرض شده است که گره‌های زوج گره‌های متخاصم هستند که گره شماره یک را به عنوان گره هدف در نظر گرفته‌اند و قصد دارند که سطح اعتماد این گره را برای دیگر گره‌ها پایین اعلام کنند تا این گره که در اصل جزء گره‌های معمولی هست را به عنوان گره متخاصم برای دیگر گره‌های معمولی شناخته شود. برای اینکار ما نیاز به دو پارامتر داریم تا بتوانیم این کار را انجام دهیم، این پارامترها عبارتند از میزان اعتماد گره و پارامتر دوم شماره گره.

در این سند سعی شده است که تمام مراحل انجام کار را گام به گام با آوردن کدهایش به صورت عکس ارائه دهیم.

در مرحله اول در فایل `IP.h` ما به هدر بسته یک فیلد به نام `trust` مطابق شکل زیر ارائه کردیم که این عدد بیانگر میزان صلاحیت گره است



```
61 ns_addr_t dst_;
62 int ttl_;
63
64 /* Monarch extn */
65 // u_int16_t sport_;
66 // u_int16_t dport_;
67
68 /* IPv6 */
69 int fid_; /* flow id */
70 int prio_;
71
72 static int offset_;
73 inline static int& offset() { return offset_; }
74 inline static hdr_ip* access(const Packet* p) {
75     return (hdr_ip*) p->access(offset_);
76 }
77
78 /* per-field member access functions */
79 ns_addr_t& src() { return (src_); }
80 nsaddr_t& saddr() { return (src_.addr); }
81 int32_t& sport() { return src_.port; }
82
83 ns_addr_t& dst() { return (dst_); }
84 nsaddr_t& daddr() { return (dst_.addr); }
85 int32_t& dport() { return dst_.port; }
86 int& ttl() { return (ttl_); }
87 /* ipv6 fields */
88 int& flowid() { return (fid_); }
89 int& prio() { return (prio_); }
90
91 int trust;
92 };
93
94 #endif
95
```

تغییر بعدی که اعمال شده است اضافه کردن `node_id` به هر گره میباشد تا از طریق آن بتوانیم گره‌ها را از هم دیگر متمایز کنیم. برای اینکار مجبور شدیم تا از یک متغیر استاتیک استفاده کنیم که هر زمان که سازنده آن صدا زد شد یک عدد به این متغیر اضافه میکنیم و آن را به عنوان شماره آن نود در نظر میگیریم که بدین ترتیب هر نود شناسه مربوط به خودش که یکتا میباشد را داراست.

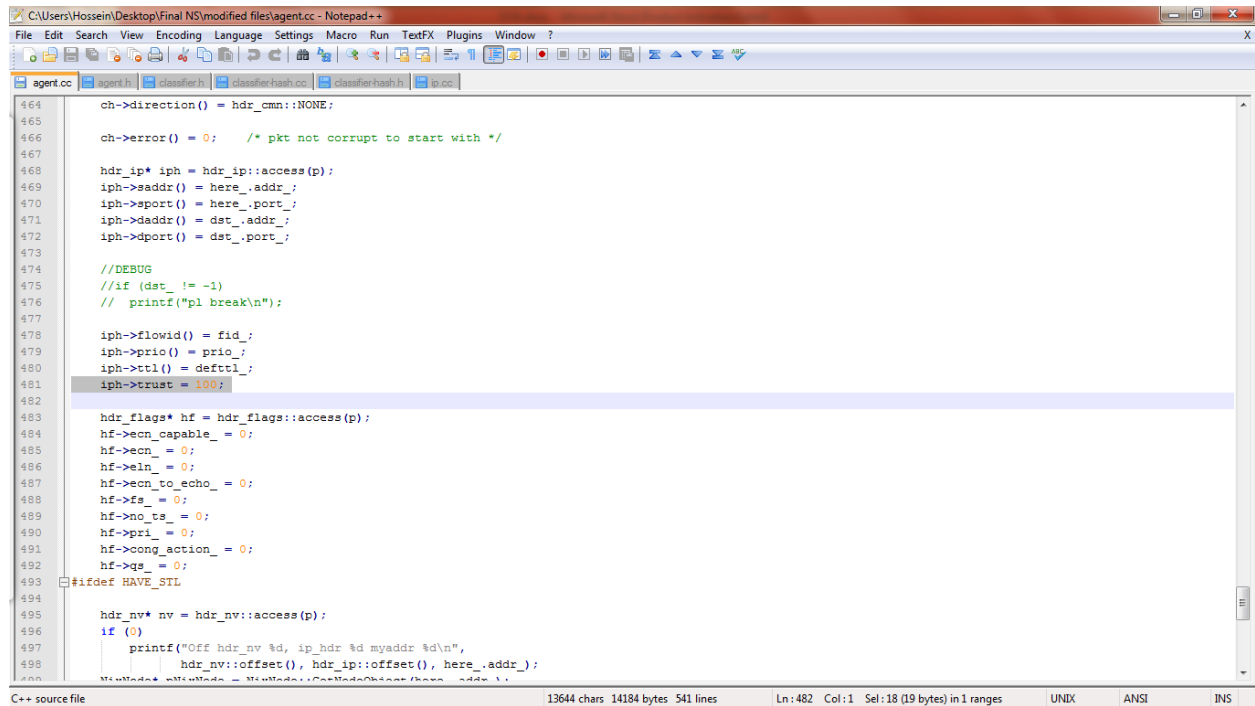
فایل classifier.h :

```
CAUsers\Hossein\Desktop\Final NS\modified files\classifier.h - Notepad++
File Edit Search View Encoding Language Settings Macro Run TextFX Plugins Window ?
agent.h classifier.h classifier-hash.cc classifier-hash.h ip.cc
23  * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
24  * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
25  * ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
26  * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
27  * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
28  * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
29  * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
30  * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
31  * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
32  * SUCH DAMAGE.
33  *
34  * @(#) $Header: /cvsroot/nsnam/ns-2/classifier/classifier.h,v 1.34 2010/03/08 05:54:49 tom_henderson Exp $ (LBL)
35  */
36
37 #ifndef ns_classifier_h
38 #define ns_classifier_h
39
40 #include "object.h"
41
42 class Packet;
43
44 class Classifier : public TObject {
45 public:
46     int node_id;
47     static int node_count;
48
49     Classifier();
50     virtual ~Classifier();
51
52     inline int maxslot() const { return maxslot_; }
53     inline TObject* slot(int slot) {
54         if ((slot >= 0) && (slot < maxslot_))
55             return slot_[slot];
56     }
57
58 };
59
60 #endif
61
62 C++ source file 3363 chars 3460 bytes 98 lines Ln:46 Col:1 Sel:41 (42 bytes) in 1 ranges UNIX ANSI INS
```

فایل classifier.cpp :

```
CAUsers\Hossein\Desktop\Final NS\modified files\classifier.cc - Notepad++
File Edit Search View Encoding Language Settings Macro Run TextFX Plugins Window ?
agent.cc agent.h classifier.cc classifier.h classifier-hash.cc classifier-hash.h ip.cc
49 int Classifier::node_count = 0;
50
51
52 static class ClassifierClass : public TclClass {
53 public:
54     ClassifierClass() : TclClass("Classifier") {}
55     TObject* create(int, const char*const*) {
56         return (new Classifier());
57     }
58 } class_classifier;
59
60 Classifier::Classifier() :
61     slot_(0), nslot_(0), maxslot_(-1), shift_(0), mask_(0xffffffff), nsize_(0)
62 {
63     default_target_ = 0;
64
65     bind("offset_", &offset_);
66     bind("shift_", &shift_);
67     bind("mask_", &mask_);
68     node_count++;
69     node_id = node_count;
70
71 }
72
73 int Classifier::classify(Packet *p)
74 {
75     return (mshift(*((int*) p->access(offset_))));
76 }
77
78 Classifier::~Classifier()
79 {
80     delete [] slot_;
81 }
82
83
84 void Classifier::Table::Classifier::
85
86 C++ source file 7495 chars 7802 bytes 308 lines Ln:70 Col:27 Sel:39 (40 bytes) in 1 ranges UNIX ANSI INS
```

حال برای مقدار دهی به این متغیر ما در فایل **agent.cc** وقتی که پکت در حال مقدار دهی اولیه است ما مقدار **trust** را در ابتدا ۱۰۰ فرض کردیم که یعنی همه نود از دیدگاه یکدیگر یکسان و از گروه نودهای معمولی به حساب بیایند.



```
464 ch->direction() = hdr_cm::NONE;
465
466 ch->error() = 0; /* pkt not corrupt to start with */
467
468 hdr_ip* iph = hdr_ip::access(p);
469 iph->saddr() = here_.addr_;
470 iph->sport() = here_.port_;
471 iph->daddr() = dst_.addr_;
472 iph->dport() = dst_.port_;
473
474 //DEBUG
475 //if (dst_ != -1)
476 // printf("pl break\n");
477
478 iph->flowid() = fid_;
479 iph->prio() = prio_;
480 iph->ttl() = defttl_;
481 iph->trust = 100;
482
483 hdr_flags* hf = hdr_flags::access(p);
484 hf->ecn_capable_ = 0;
485 hf->ecn_ = 0;
486 hf->ecn_ = 0;
487 hf->ecn_to_echo_ = 0;
488 hf->fs_ = 0;
489 hf->no_ts_ = 0;
490 hf->pri_ = 0;
491 hf->cong_action_ = 0;
492 hf->qos_ = 0;
493 #ifdef HAVE_STL
494
495 hdr_nv* nv = hdr_nv::access(p);
496 if (0)
497     printf("Off hdr_nv %d, ip_hdr %d myaddr %d\n",
498           hdr_nv::offset(), hdr_ip::offset(), here_.addr_);
499
500
```

و در نهایت به فایل‌های **classifier-hash.h** و **classifier-hash.cpp** می‌رسیم که کارهای اصلی را در این قسمت انجام شده است.

ابتدا در هدر این کلاس آرایه‌ای از میزان اعتماد خود نسبت به دیگران را به نام **n_trust[]** در نظر می‌گیریم و مقدار دهی اولیه می‌کنیم:

```
C:\Users\Hossein\Desktop\Final NS\modified files\classifier-hash.h - Notepad++
File Edit Search View Encoding Language Settings Macro Run TextFX Plugins Window ?

classifier-hash.h classifier-hash.h

155     buf_.SrcDst.dst= mshift(dst);
156     return (const char*) &buf_;
157 }
158 };
159
160 class FidHashClassifier : public HashClassifier {
161 public:
162     FidHashClassifier() : HashClassifier(TCL_ONE_WORD_KEYS) {
163     }
164 protected:
165     const char* hashkey(nsaddr_t, nsaddr_t, int fid) {
166         long key = fid;
167         return (const char*) key;
168     }
169 };
170
171 class DestHashClassifier : public HashClassifier {
172 public:
173     DestHashClassifier() : HashClassifier(TCL_ONE_WORD_KEYS) {
174         for(int i=0;i<60;i++)
175             n_trust[i]=90;
176     }
177     int n_trust[60];
178     virtual int command(int argc, const char*const* argv);
179     int classify(Packet* p);
180     virtual void recv(Packet* p, Handler* h);
181     virtual void do_install(char *dst, NsObject *target);
182
183 protected:
184     const char* hashkey(nsaddr_t, nsaddr_t dst, int) {
185         long key = mshift(dst);
186         return (const char*) key;
187     }
188 };
189
C++ source file 5477 chars 5667 bytes 191 lines Ln:174 Col:5 Sel:56 (59 bytes) in 1 ranges UNIX ANSI INS
```

سپس در فایل **cpp** . آن، در جایی که پکت دریافت میشود یعنی در تابع **recv(packet*,Handler*)** کار اصلی یعنی تغییر میزان اعتماد گره‌ها بر اساس اینکه گره معمولی میباشد یا گره متخصص انجام میشود که در ابتدا چک میشود که آیا این پکت از طرف گره قابل اعتمادی هست یا نه که اگر این مقدار کمتر از ۱۰ باشد ما این گره را به عنوان گره غیر قابل اعتماد میدانیم و آن بسته را دور میریزیم (drop میکنیم) و سپس آرایه‌ای که مربوط به میزان اعتمادی که به دیگران دارد را به‌روزرسانی میکنیم به این ترتیب که میانگین وزنی گرفتیم و میزان اعتمادی که قبلاً داشته را با وزن ۹۹۹ در نظر میگیریم و میزان اعتمادی که در بسته اعلام شده است را با وزن ۱ در نهایت به گره‌های متخصص میرسیم که این گره‌ها یعنی گره‌های زوج همانطور که قبلاً بیان شد قصد دارند که گره با شماره یک را به عنوان متخصص جلوه دهند و ازین رو **trust** آن را برابر صفر قرار میدهند.

```
C:\Users\Hossein\Desktop\Final NS\modified files\classifier-hash.cc - Notepad++
File Edit Search View Encoding Language Settings Macro Run TextFX Plugins Window ?

classifier-hash.cc classifier-hash.h

223 void DestHashClassifier::recv(Packet* p, Handler* h){
224     hdr_ip* iph = hdr_ip::access(p);
225     na_addr_t pp = iph->src();
226     cout << "my name : "<<((node_id+1)/2)<<"\n";
227     int myID = ((node_id+1)/2) - 1;
228     int32_t packet_ip = pp.addr_;
229     cout << "packet source IP : "<< packet_ip;
230     cout << " ** packet trust : "<< iph->trust<<"\n";
231     static long int drops=0;
232     if(n_trust[ packet_ip ] < 10)
233     {
234         Packet::free(p);
235         drops++;
236         cout << " Me: "<<myID<< " DROPS: "<<drops<<"\n";
237         //if(myID<3){cout<<"???";}
238         return;
239     }
240     // All
241     cout << " Trust["<<(packet_ip)<<"] Old= "<<n_trust[ packet_ip ];
242     int temp = n_trust[ packet_ip ];
243     float result = ( 999*temp + iph->trust ) / 1000 ;
244     n_trust[ packet_ip ] = int(result);
245     if(n_trust[ packet_ip ] < 10 ){
246         cout <<myID<< " DROPS: "<<drops<<"\n";
247         //exit(1);
248     }
249     if(n_trust[ packet_ip ] < 0)
250         n_trust[ packet_ip ] = 0;
251     cout << " ** New= "<<n_trust[ packet_ip ];
252     // I'm evil
253     if ( myID % 2 == 0 && (packet_ip) == 1 ) {
254         cout << "... my name : "<<myID<<"\n";
255         iph->trust = 0 ;
256     }
257     Classifier::recv(p,h);
258     return;
259 }

C++ source file 8383 chars 8647 bytes 265 lines Ln: 240 Col: 11 Sel: 0 (0 bytes) in 0 ranges UNIX ANSI INS
```

نتیجه گیری:

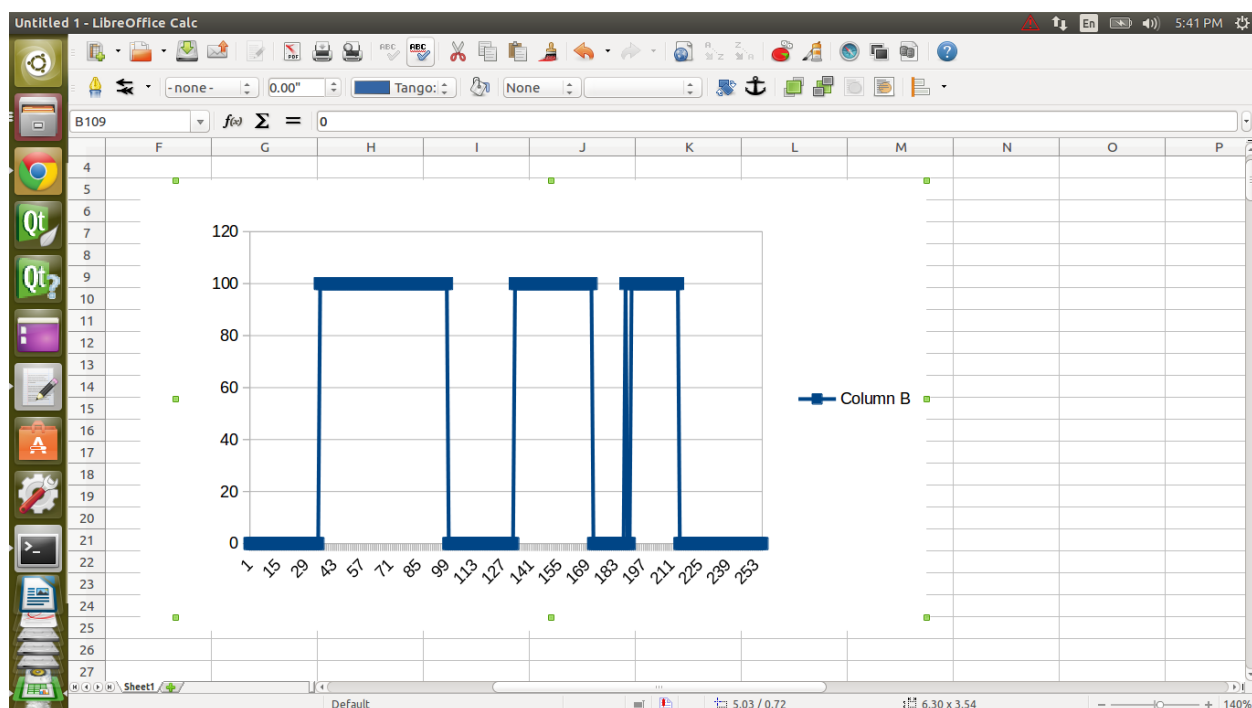
در نهایت برای نشان دادن نتیجه کارمان تصمیم گرفتیم که نمودارهایی ارایه دهیم که در زیر آورده شده است.

نمودار اول :

محور عمودی: میزان trust درج شده برای گره ۱

محور افقی: شماره بسته (شماره بسته‌ی رسیده به گره جاری با مبدا گره ۱)

در این نمودار میزان trust گرهی ۱ رسیده به یک گره تصادفی را نشان میدهد. از آنجا که ما دو حالت $trust=100$ (پیشفرض) و $trust=0$ (تغییر داده شده توسط گره های متخاصم) نداریم، نمودار این دو عدد را بعنوان trust درج شده در بسته های رسیده به وی (با فرض مبدا از گره شماره ۱) نشان میدهد.



نمودار دوم :

محور عمودی: میزان trust فعلی گره ۱ از نگاه گره جاری (یک گره تصادفی)

محور افقی: شماره بسته (شماره بسته‌ی رسیده به گره جاری با مبدا گره ۱)

در این نمودار ما میزان trust ثبت شده در جدول برای گره ۱ را از نگاه گره جاری در طی زمان نشان داده ایم. این جدول بصورت پیشفرض با مقدار ۹۰ مقداردهی اولیه میشود و بر اساس بسته های نمایش داده شده در نمودار ۱ و با توجه به وزن دهی توضیح شده در قبل ، بروزرسانی میشود.

