EEL6951 Project: Simulation of "LDoS Attack in Ad-hoc Network" by Y. He et al.

By Paul Muri, Carlo Pascoe, and Brian Sapp

I. Introduction

Research Paper: "LDoS Attack in Ad-hoc Network" by Y. He et al.

Citation: Yanxiang He; Yi Han; Qiang Cao; Tao Liu; Libing Wu, "LDoS attack in ad-hoc network," Wireless On-Demand Network Systems and Services, 2009. WONS 2009. Sixth International Conference on , vol. 1, no., pp.251-257, 2-4 Feb. 2009 (7pages)

Abstract: LDoS (Low-rate Denial of Service) attack is periodic, stealthy, and with high efficiency, which has become a great threat to the network security. Previous researches about LDoS attack mainly focus on its impact on wired networks. However, our analysis shows that such attack could also be launched in Ad-hoc network, and as a completely distinct MAC layer protocol is adopted in this environment, the form and effect of the attack could be different and need re-evaluation. This paper presents a study of LDoS attack in Ad-hoc network: (1) We investigate the differences of attack form brought by the medium reservation mechanism and CSMA/CA of 802.11b, and find that decreasing the period of LDoS attack into a smaller time scale would achieve a higher attack efficiency; (2) We show that the attack effect differs from that in wired networks, and the attacker's location has an impact on it; (3) We verify our findings by simulation experiments in ns-2; (4) Detection and defense methods are explored to counter against such attack.

URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4801858&isnumber=4801825

Definition of Selected Topic

While previous research has gone into low-rate denial of service (LDoS) attacks on wired networks, our paper addresses the analysis of this type of attack in an ad-hoc network. By sending short bursts periodically, LDoS attacks exploit the Transmission Control Protocol (TCP) congestion deficiencies. As a result, LDoS attacks could also be done in an ad-hoc network. However, since an ad-hoc network has a different MAC layer protocol, the effect of the attack will be different. First, the paper states that with a medium reservation

mechanism and carrier sense multiple access with collision avoidance (CSMA/CA) in 802.11b, decreasing the period of LDoS attack into smaller time scales would result in a more severe attack. Second, the paper shows that an attack effect is different from wired network because attackers can be at different locations. The paper verifies this with simulations. Last, the paper suggests detection and defense methods to counter an attack.

Our project will simulate an LDoS attack on wireless ad-hoc network. For ns-2 we are going to use the dynamic source routing (dsr) protocol, and a standard 802.11 MAC protocol. Specifically, the paper says CSMA/CA in 802.11b. We shall simulate and analyze the impact of varying the attack period (T), burst duration (t), amplitude (R), contention window minimum (CWmin), and location of attacker.

Paper's Architecture

The paper's network architecture shows four nodes. The nodes are labeled as 0, 1, 2, and 3. Node 0 and 1, Node 1 and 2, Node 1 and 3 are all 200 meters apart. The carrier sense distance of each node is 550 meters and the data transmission distance is 250 meters. The paper uses a MAC layer 802.11b protocol, so the data transmission rate is 11 Mbps and the route protocol is dsr. The legal sender is Node 0 which sends TCP packets to Node 2. Node 1 is a forwarding node. Node 3 is an attacker, which floods Node 1 with UDP packets. Node 1 forwards the UDP packets to Node 2. The attack causes a packet loss for the TCP packets. Figure 1 shows the LDoS attack model in the paper while figure 2 shows the LDoS attack model created in ns2 run on the network animator (nam) below.

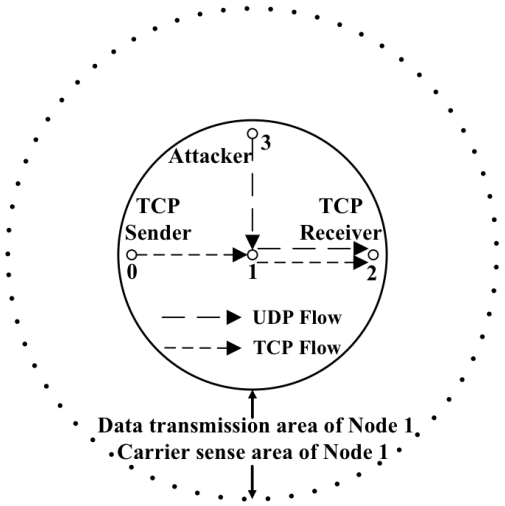

Figure 1 LDoS attack model from the paper

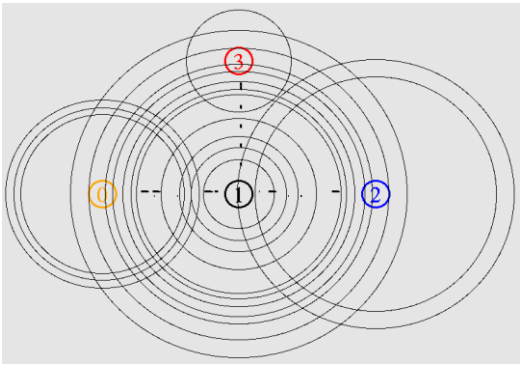

Figure 2 LDoS attack model from ns2 simulation shown in the NAM

Paper's Algorithm

The paper's network algorithm considers the MAC layer's uniqueness in an Ad-hoc network simulation. The measurement algorithm is based on the variance of jitter and throughput of request to send (RTS) frames. A TCP-targeted LDoS attack is launched by the attacker sending short duration, high-rate burst at the time attack period (T), which causes all legal TCP packets to be dropped. So, a good TCP sender will have to wait for the retransmission timer to expire and multiply its retransmission timeout (RTO) by a constant known as q. q is typically two and when q is two it is known as a binary exponential back-off. So, the retransmission timeout equation would be RTO[K+1] = q × RTO [K]. So, the legal TCP senders retransmit their lost packets at time T + RTO, which is the same time the attacker sends a burst. The same occurs at T + 3*RTO, T + 7*RTO, T + 15*RTO, … , the attacker can deny service to legal TCP flows even with a low rate or large period.

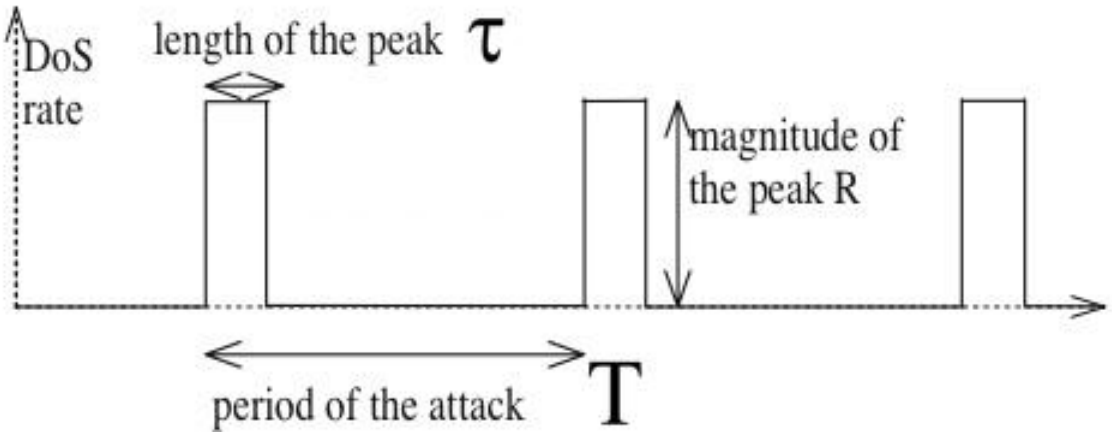

Figure 3 An example of a square-wave LDoS stream

II. Reference Paper's Algorithm

Topologies

This paper presented an analysis of LDoS attacks in an Ad-hoc network through simulation experiments in NS-2. The experiment is setup as follows. Three mobile nodes are stationary in an Ad-hoc network with node 0 200m from node 1, node 1 200m from node 2, and node 2 400m from node 0. Each node has a carrier sense distance of 550m and a data transmission distance of 250m (defualt values). By use of the DSR protocol it is established that node 1 can transmit to either 0 or 2, but nodes 0 and 2 can only communicate with each other by forwarding packets through 1. Node 0, fully adhering to the 802.11b MAC protocol,

begins sending an endless stream of tcp packets to node 2, through node 1, achieving some throughput in Mb/s. An attacker node, positioned 200m from node 1 and 283m from nodes 0 and 2, begins transmitting udp packets to node 2, through node 1, while only loosely following the 802.11b MAC protocol in a malicious attempt to disrupt the connection that node 0 has established with node 2, the inevitable result being a reduction in the realized throughput from node 0. A diagram of this setup can be seen in Figure 4 of the paper (reproduced in Figure 1 of this report). The frequency, duration, and severity of these attacks are variables in the experiment, with T defining the attack period, t defining the attack duration, and R defining attack amplitude, the transmission rate of the udp packets during an attack duration. The remainder of this section discusses how the authors vary these attack parameters, along with contention window parameters, in order to fully assess the effects of LDoS attacks on such an Ad-hoc network.

Attack period (T) Effects

The goal of a LDoS attack is to degrade the tcp performance of a valid connection; the effect of such attacks is to minimize the throughput of a tcp connection by causing the dropping of packets. Unlike the medium reservation mechanisms available in wired networks, the wireless network under discussion here requires the exchange of rts and cts frames prior to the actual data frame; unlike wired networks, LDoS attacks in this wireless network mostly cause the dropping of rts and cts packets rather than data packets. In order to maximize the number of packets dropped, the attack period should be chosen in order to minimize the sender's cwnd (ie T should be shorter than the recovery time of the sender's cwnd which gets decreased due to the last LDoS burst). The authors' analysis shows that decreasing the period of LDoS attacks into smaller time scales will achieve higher attack efficiency because it will cause more frequent disruption of the channel and therefore more dropped packets.

Burst duration/Period and Amplitude Effect

In order to keep the low-rate characteristic of an LDoS attack the burst duration and attack amplitude should be made large enough to cause the loss of tcp packets but as small as possible to minimize the amount of attacker traffic. The authors propose that as the attack burst duration moves closer to the attack period in a ratio of t/T, the attack efficiency increases by reducing the normalized throughput (though not properly defined in the paper, we assume to be the ratio of the throughput under attack to the throughput not

under attack) but only to a limit of 0.5; as the ratio increases past 0.5, attack efficiency is not increased significantly and the increased traffic does not conform to the low-rate characteristic. The attack amplitude needs to be larger than the achieved throughput of the tcp connections under attack so that it is high enough to overwhelm the legitimate tcp flows. The authors' analysis shows that increasing the attack amplitude improves attack efficiency, but only to a point (when the attacker's rate matches the achieved throughput of the tcp connection) after which performance remains almost constant. Although the transmission rate of this experiment can reach theoretically reach 11Mb/s according to 802.11b, in real scenarios the actual throughput of a tcp connection is far below that value, so the amplitude of a LDoS attack need not be even close to 11Mb/s (for this experiment the sweet spot is shown to be 2Mb/s).

Simulation Experiments

The body of the paper is centered on four separate experiments that analyze the impact of an LDoS attack on an Ad-hoc network. The experiments are titled as follows: Impact of attack period and burst duration on attack Effect, Impact of attack amplitude on attack effect, Impact of CWMin and CWMax on LDoS stream, and Impact of attacker's location on attack effect.

Impact of Attack Period and burst duration on attack Effect

In the set of experiments assessing impact of attack period and burst duration on attack effect, the amplitude of the LDoS stream is held constant at 2Mb/s, the attack period is varied from 0.05s to 2.0s, and the burst duration is proportional to the period using several t/T constant values ranging from 0.1 to 0.5. It is observed that with the same proportion, as the attack period decreases, the normalized throughput diminishes significantly as one would expect given the analysis previously discussed. These experiments are extended further to record the cwnd of the victim node under LDoS attacks for different periods and burst durations. It is observed that when the period decreases from 2.0s to 0.5s, cwnd of the victim node decreases as anticipated, but only to a point and even reverses as the period further decreases to 0.05s. This phenomenon can be explained because if the burst duration is too short, legal tcp packets won't be dropped at every LDoS burst.

Impact of attack amplitude on attack effect

In the two experiments investigating the impact of attack amplitude on attack effect, the attack periods are held constant at 0.1s and 0.5s, and the burst durations are held constant at 0.02s and 0.1s, while the attack amplitude is varied from 0.2Mb/s to 11Mb/s. The normalized throughput (which we assume to be the throughput under attack divided by the throughput not under attack) decreases as the amplitude increases, but almost maintains the same value when the amplitude is increased above 2Mb/s. As discussed earlier this is because at about the 2Mb/s the attack throughput matches the victim nodes throughput and anything above this has no effect.

Impact of CWMin and CWMax on LDoS stream

In the set of experiments investigating the impact of CWMin and CWMax on a LDoS stream the parameters attack period of 1s, attack duration of 0.2s, and attack amplitude of 2Mb/s are held constant while the attacker's CWMin and CWMax are varied. When the channel is idle for a DIFS, each node still has to "generate a random backoff period for an additional deferral time before transmitting, unless the backoff timer already contains a nonzero value" causing attack packets to not be sent out at the scheduled time (this distorts the expected square-wave shape of the LDoS stream). If the minimum and maximum values of the attacker's contention window are altered to be smaller than those of other legal users, attack packets will essentially have priority over non attack packets, increasing the effectiveness of the attack. It is observed in this experiment that the LDoS stream is mainly affected by the value of CWMin.

Impact of attacker's location on attack effect

The final set of experiments analyzes the impact of attacker's location on attack effect. The results of these experiments are as follows: there is little attack effect if the attacker is located outside of the carrier sense areas of the other nodes. Also, forwarding nodes severely change the pattern and amplitude of the LDoS stream because of delays and drop packets. The LDoS is supposed to look like a square wave if directly attacking, no hops between the attacking and victim node. By introducing intermediate nodes in the middle to forward packets, delays and drop packets distort the square wave.

III. The Team's Algorithm

Our algorithm effectively implements and improves the work of the original authors. It simulates a four node wireless ad hoc network in a 1000m by 1000m flat topography. The ad hoc routing protocol is DSR and the MAC protocol is the antiquated 802.11b protocol. Our script is configured to accept input arguments of attack period, attack duration, attack amplitude and optionally the minimum and maximum 802.11b contention windows. It also reports the average TCP congestion window, the normalized throughput, and the average effective burst ratio. We additionally implemented a shell script to automatically call NS2 and pass along the necessary input arguments to simulate all the scenarios described by the paper.

After defining the global variables and options, our algorithm implements the legal and attack traffic patterns. Node 0 is associated with a TCP agent with FTP traffic. Node 0 transmits the data to Node 2 via Node 1. Node 2 is linked to a TCP sink that sends one ACK packet to Node 0, via Node 1, per TCP packet received. The attacker, Node 3, transmits CBR packets to Node 2 via Node 1 beginning at 10.0s and ending at 40s. The total simulation time is 50s per scenario.

We use a set of perl and awk scripts, coupled with the shell script, to extract the average TCP congestion window, the normalized throughput, and the average effective burst ratio.

Normalized Throughput discussion

Understanding how the authors in the paper simulated the effect of t/T was straightforward. The T and t variable could be set up from on the attack.tcl script script. However, the author's definition of throughput was puzzling. For instance, in figure 8 of the paper the normalized throughput is at one. However, "Communication Networks" by Leon Garcia defines normalized throughput as "the actual rate at which information is transmitted through the shared medium. The throughput is measured in [frames/second] or [bits/second]. Suppose that the transmission rate in the medium is R [bits/second] and suppose that the average frame length is L [bits/frames]. The maximum possible throughput is then R/L [frames/second]." Since some channel time is wasted in collisions or in sending coordination information each MAC protocol has a maximum throughput less than R/L. This is especially true in a wireless network. So, the fact that the paper could state

a normalized throughput of 1 is not correct. So, we had to look at another way the definite normalized throughput.

We found that the authors calculated normalized throughput as the ratio of throughput in bytes per second when not under attack to the throughput during the attack. Explicitly, it is calculated as the total number of tcp bytes sent at the MAC layer by the legal sender, Node 0, during times 0-10s and 40-50s divided by the total number of tcp bytes at the MAC layer by Node 0 during the attack period (10-40s).

$$normalized\,throughput = \frac{tcp\,throughput_{attack}}{tcp\,throughput_{noattack}}$$

It should be noted that the authors deviated from the established CSMA/CA of 802.11b by setting the minimum and maximum contention windows from 1023 and 31, respectively to much smaller values. This is to prevent attack packets from competing for the medium. In our simulation, we set CWmax and CWmin both to 1 for this series of data and found the results nearly duplicated the work by the authors. We propose that it would be more accurate to set only the CWmax and CWmin of the attack node(s) instead of all nodes.

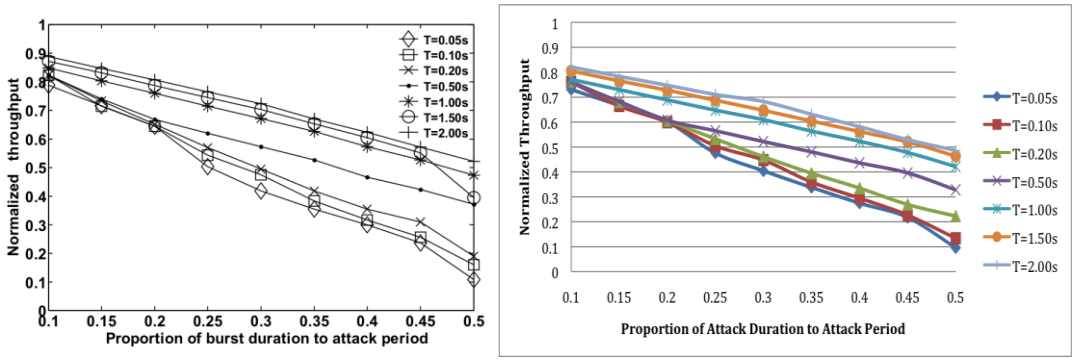

Figure 4 Impact of Attack period and burst duration on normalized throughput

TCP Congestion Window

The TCP protocol uses a congestion window to determine how many bytes can by outstanding at any time. This is used to prevent the network from being overloaded. There are different algorithms to calculate the window depending on the specific tcp protocol variation.

We found after careful study that the authors did not measure the average congestion window directory. Instead, they measured the slow-start threshold of the tcp agent while incorrectly labeling the results as the average congestion window. The slow-start threshold is closely related the congestion window, so the results are still very close.

Our algorithm uses an internal state variable of the TCP agent to report the average congestion window at the end of each simulate scenario. Once again we set CWmax and CWmin both to 1 for this series of data.

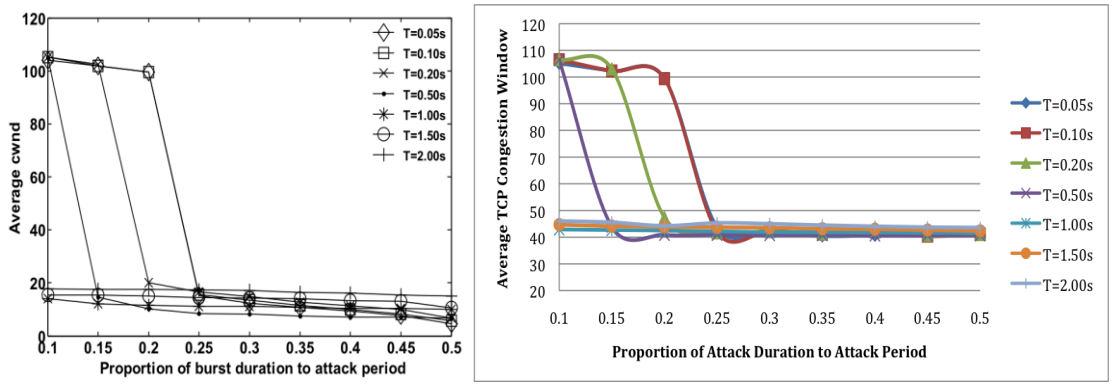

Figure 5 Impact of attack period and burst duration on congestion window

Attack Amplitude

The term attack amplitude used by the authors is a misnomer. Our algorithm more accurately describes the LDoS parameter as the attack rate. Specifically, we set Node 3 to send cbr packets of size 512B at 2Mbs where 2Mb/s is the attack rate.

We found that there is almost no change in the affect of the attack for rates above 2Mb/s. While the theoretical throughput is near 11Mb/s, the observed typical throughput in an 802.11b environment is less than 5Mb/s.

It is interesting to note that the authors chose not to use a CWMin and CWMax of 1 as in previous cases to study the affect of the attack amplitude. We observed that the results were obtained with standard values of 31 and 1023 respectively after simulating both with our algorithms.

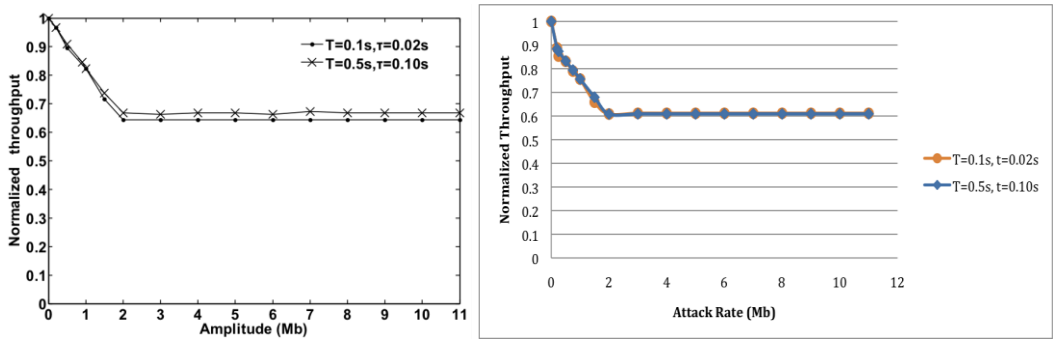

Figure 6 Impact of amplitude on normalized throughput

802.llb Contention Window and Effective Average Attack Duration

The default values of CWMax and CWmin for 802.11b are 1023 and 31 respectively. We varied both values while measuring the effective average burst duration.

The authors did not define "real burst duration" so we had to investigate the trace files to accurately study the metric. As anticipated, we found that although we specified specific attack starts (relative to the static variables attack period and attack duration) that the attack effect continued due to traffic congestion, retransmissions, and other simulated real-world factors.

The "real burst duration," more accurately termed the average effective attack duration, represents the difference between when an attack started and when an attack stopped within an attack period. We calculated the values by observing the cbr traffic in the trace files.

Our results for a period of 1s, attack duration of 2s, and an attack rate of 2Mb/s are below.

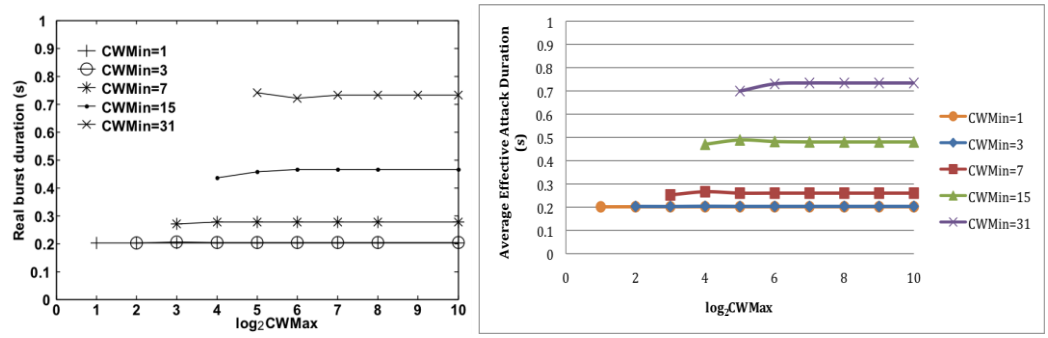

Figure 7 Impact of CWmin and CWmax on real burst duration

Attacker Locations

Blah blah blah…

IV. Conclusion

Discussion of NS2

In our paper's abstract is read that they "verify [their] findings by simulation experiments in NS-2". Since this project required each group to find a paper, which used NS2 to simulate, we choose this paper and NS2 as our tool for simulation. NS2 is a very convenient tool to simulate wired and wireless network because it saves on the hardware needed to actually produce the network. Also, it is flexible in changing the network very quickly. Some of the results obtained from NS2 simulations cannot be measured in a real world network as well.

Difficulties in the ns-2 implementation of the algorithm

The biggest difficulty was figuring out how the paper failed to define variables such as normalized throughput, real burst duration, and congestion window.

The definition discrepancy of normalized throughput was mentioned in the team analysis section of this report. As an improvement to this paper we able the figure out the real normalized throughput from a definition in Leon-Garcia's textbook and present those results.

Also, the paper we used failed to define results of real burst duration. After some research and consulting the author we found that it was the last UDP packet sent during one period (T) of an attack. For example, if T=1 second and tau=0.2 seconds it could be that because of collisions and the minimum contention windows that the last UDP is at 0.7 seconds. Thus the real burst duration is 0.7 seconds instead of 0.2 seconds. After this was figured it out it was real easy to write an awk script to average all the real burst durations together for all the attack periods down.

The paper also had a graph we wanted to reproduce showing the congestion window. The graph trend looked strange because it just cut off after to an average congestion window of 20 after the t/T ratio was greater than 1/4. After taking a look at how NS-2 produced an average congestion window straight from the .tcl simulation we decided this variable would be much more accurate than the script the authors may have wrote to average all the congestion windows together. We learned from using the average that the average congestion window should settle around 40 slots.

Final Division of Labor

This project consisted of creating a tcl file that modeled the situations described in the paper. In the proposal, the creation of the tcl file will be handled as follows:

1. Paul will analyze the attack period and burst duration on attack effect.

2. Carlo will analyze the impact of attack amplitude on attack effect.

3. Brian will analyze the impact of CWMin and CWMax on the LDoS stream, and also analyze the impact of attacker's location on the attack effect.

Towards the end, a more synergized effort was used for tackle how to simulate each of these sections. The initial .tcl coding was done as a group. Carlo and Paul worked more in depth on awk scripts for finding the normalized throughput and real burst duration. Brian found an agent to obtain a more accurate congestion window value than used in the paper. Brian kept working on improving the .tcl code, automating simulations using a shell script, parsing the results with a perl script, and generating the graphs.

NS-2 Library files used

The relevant ns-2 library files for our proposal are listed below.

lnx16:/apps/ns/ns-2.33/dsr

dsragent.cc, dsragent.h, dsragent.o, dsr_proto.cc, dsr_proto.h, dsr_proto.o

lnx16:/apps/ns/ns-2.33/mac

mac-802_11.cc, mac-802_11.h, mac-802_11.o, wireless-phy.cc, wireless-phy.h,

wireless-phy.h, channel.cc, channel.h, channel.o

References

Leon-Garcia, Computer Networks, 2 ed. , New York: McGrall-Hill, 2000.

X Graph http://www.xgraph.org/

Perl scripting to a file http://perl.about.com/od/perltutorials/a/readwritefiles.htm

http://nile.wpi.edu/NS/

http://www.isi.edu/nsnam/ns/tutorial/

http://www.ece.uvic.ca/~emads/ns-2.29/class_trace_file.html

http://www.tcl.tk/man/tcl/tutorial/tcltutorial.html

http://users.belgacom.net/bruno.champagne/tcl.html

http://www.grymoire.com/Unix/Awk.html

http://www.softpanorama.org/Tools/awk.shtml

http://www.comptechdoc.org/independent/web/cgi/perlmanual/

http://www.linuxconfig.org/Bash_scripting_Tutorial

Appendix

Throughput.awk

```
BEGIN {
        i = 10;
        j = 0.0;
        anum = 0;
        num = 0;
        lastPID = -1;
        dup = 0;
}

{
        strEvent = $1 ;
        rTime = $2 ;
        nodenum = $3 ;
        strAgt = $4 ;
        idPacket = $6 ;
        strType = $7 ;
        bytes = $8 ;
        Pid = $18 ;

                if ( strEvent == "s" && nodenum == "_0_" && strAgt == "MAC"   &&   strType
== "tcp" ){
        #printf("%s, %d\n", substr(Pid,2, length(Pid)),substr(Pid,2, length(Pid)));
        if( substr(Pid,2, length(Pid)) == last+1){
                last = last + 1;

                        if( int( rTime) > 10 && int( rTime) < 40){


                                anum = anum + bytes;
                        }else{
                                num = num + bytes;
                        }
        }else{
                dup = dup + 1;
        }
                }
}
END {
        printf( "Throughput: %3.5f, %d\n", (anum*2)/(num*3), dup) ;
}
```

Realburst.awk

```
BEGIN {
        T = 1;
        i = 10+T;
        j = 0.0;
        total = 0.0;
        num = 0;
}

{
        strEvent = $1 ;
        rTime = $2 ;
        nodenum = $3 ;
        strAgt = $4 ;
        idPacket = $6 ;
        strType = $7 ;
        bytes = $8 ;

        if ( strEvent == "s" && nodenum == "_3_" && strAgt == "MAC"   &&   strType ==
"cbr" ) {
                if(rTime > i ){
                        total = total + j - i + T;
                        num = num + 1;
                        printf("%f, %f\n", j, i-T);
                        i = i+T;
                }
                j = rTime;
        }
}


END {
                        total = total + j - i + T;
                        num = num + 1;
                        printf("%f, %f\n", j, i-T);
        printf( "Real Burst Duration: %3.5f\n, %d, %d %f \n", total/num, total, num, j  )
;
}
```

T	t	Am	Cwmin	Cwmax	t/T	Th	cwnd
0.05	0.005	2	1	1	0.1	0.73038	105.15
0.05	0.0075	2	1	1	0.15	0.66642	102.3
0.05	0.01	2	1	1	0.2	0.60038	99.45
0.05	0.0125	2	1	1	0.25	0.47567	43.76
0.05	0.015	2	1	1	0.3	0.40413	42.54
0.05	0.0175	2	1	1	0.35	0.3377	40.5
0.05	0.02	2	1	1	0.4	0.27461	40.74
0.05	0.0225	2	1	1	0.45	0.2171	40.76
0.05	0.025	2	1	1	0.5	0.09589	41.07
0.1	0.01	2	1	1	0.1	0.76034	106.48
0.1	0.015	2	1	1	0.15	0.66451	102.24
0.1	0.02	2	1	1	0.2	0.59918	99.39
0.1	0.025	2	1	1	0.25	0.50463	42.08
0.1	0.03	2	1	1	0.3	0.44703	42.42
0.1	0.035	2	1	1	0.35	0.35919	41.27
0.1	0.04	2	1	1	0.4	0.29477	42
0.1	0.045	2	1	1	0.45	0.22646	40.42
0.1	0.05	2	1	1	0.5	0.13374	41.15
0.2	0.02	2	1	1	0.1	0.76186	106.48

0.2	0.03	2	1	1	0.15	0.68048	103.01
0.2	0.04	2	1	1	0.2	0.60584	47.4
0.2	0.05	2	1	1	0.25	0.53247	41.61
0.2	0.06	2	1	1	0.3	0.46093	42.13
0.2	0.07	2	1	1	0.35	0.39457	41.2
0.2	0.08	2	1	1	0.4	0.33481	43.47
0.2	0.09	2	1	1	0.45	0.26921	40.77
0.2	0.1	2	1	1	0.5	0.22248	40.98
0.5	0.05	2	1	1	0.1	0.75993	106.41
0.5	0.075	2	1	1	0.15	0.6824	43.95
0.5	0.1	2	1	1	0.2	0.60736	40.9
0.5	0.125	2	1	1	0.25	0.56482	40.89
0.5	0.15	2	1	1	0.3	0.52167	40.79
0.5	0.175	2	1	1	0.35	0.48019	40.78
0.5	0.2	2	1	1	0.4	0.43617	40.64
0.5	0.225	2	1	1	0.45	0.39571	40.63
0.5	0.25	2	1	1	0.5	0.32798	40.72
1	0.1	2	1	1	0.1	0.7709	42.91
1	0.15	2	1	1	0.15	0.72977	42.7
1	0.2	2	1	1	0.2	0.68852	42.5
1	0.25	2	1	1	0.25	0.64757	42.19
1	0.3	2	1	1	0.3	0.60941	41.88
1	0.35	2	1	1	0.35	0.56366	41.84
1	0.4	2	1	1	0.4	0.52271	41.77
1	0.45	2	1	1	0.45	0.4777	41.67
1	0.5	2	1	1	0.5	0.42116	41.47
1.5	0.15	2	1	1	0.1	0.80518	44.69
1.5	0.225	2	1	1	0.15	0.76503	44.18
1.5	0.3	2	1	1	0.2	0.72757	43.88
1.5	0.375	2	1	1	0.25	0.68732	43.71
1.5	0.45	2	1	1	0.3	0.64668	43.52
1.5	0.525	2	1	1	0.35	0.60405	43.16
1.5	0.6	2	1	1	0.4	0.56174	42.98
1.5	0.675	2	1	1	0.45	0.51998	42.67
1.5	0.75	2	1	1	0.5	0.46308	42.35
2	0.2	2	1	1	0.1	0.82081	46.07
2	0.3	2	1	1	0.15	0.78329	45.56
2	0.4	2	1	1	0.2	0.74728	44.26
2	0.5	2	1	1	0.25	0.71057	45.34
2	0.6	2	1	1	0.3	0.68202	45.01
2	0.7	2	1	1	0.35	0.63108	44.49
2	0.8	2	1	1	0.4	0.58128	44.07
2	0.9	2	1	1	0.45	0.52912	43.75
2	1	2	1	1	0.5	0.48585	43.6

0.1	0.02	0	31	1023	0.2	1
0.1	0.02	0.2	31	1023	0.2	0.88754
0.1	0.02	0.25	31	1023	0.2	0.85333
0.1	0.02	0.5	31	1023	0.2	0.8311
0.1	0.02	0.75	31	1023	0.2	0.78981
0.1	0.02	1	31	1023	0.2	0.75608
0.1	0.02	1.5	31	1023	0.2	0.65857
0.1	0.02	2	31	1023	0.2	0.60844
0.1	0.02	3	31	1023	0.2	0.61151
0.1	0.02	4	31	1023	0.2	0.61151
0.1	0.02	5	31	1023	0.2	0.61151
0.1	0.02	6	31	1023	0.2	0.61151
0.1	0.02	7	31	1023	0.2	0.61151
0.1	0.02	8	31	1023	0.2	0.61151
0.1	0.02	9	31	1023	0.2	0.61151
0.1	0.02	10	31	1023	0.2	0.61151
0.1	0.02	11	31	1023	0.2	0.61151
0.5	0.1	0	31	1023	0.2	1
0.5	0.1	0.2	31	1023	0.2	0.88332
0.5	0.1	0.25	31	1023	0.2	0.87334
0.5	0.1	0.5	31	1023	0.2	0.8325
0.5	0.1	0.75	31	1023	0.2	0.79428
0.5	0.1	1	31	1023	0.2	0.75618
0.5	0.1	1.5	31	1023	0.2	0.67926
0.5	0.1	2	31	1023	0.2	0.60847
0.5	0.1	3	31	1023	0.2	0.60904
0.5	0.1	4	31	1023	0.2	0.60904
0.5	0.1	5	31	1023	0.2	0.60904
0.5	0.1	6	31	1023	0.2	0.60904
0.5	0.1	7	31	1023	0.2	0.60904
0.5	0.1	8	31	1023	0.2	0.60904
0.5	0.1	9	31	1023	0.2	0.60904
0.5	0.1	10	31	1023	0.2	0.60904
0.5	0.1	11	31	1023	0.2	0.60904

Cwmin	Cwmax	log2(realburst)	Real burst
31	32	5	0.69949
31	64	6	0.73069
31	128	7	0.7345
31	256	8	0.7345
31	512	9	0.7345
31	1024	10	0.7345
15	16	4	0.47082
15	32	5	0.48928

15	64	6	0.48259
15	128	7	0.48071
15	256	8	0.48071
15	512	9	0.48071
15	1024	10	0.48071
7	8	3	0.25287
7	16	4	0.26665
7	32	5	0.26068
7	64	6	0.26068
7	128	7	0.26068
7	256	8	0.26068
7	512	9	0.26068
7	1024	10	0.26068
3	4	2	0.20353
3	8	3	0.2032
3	16	4	0.20415
3	32	5	0.20359
3	64	6	0.20359
3	128	7	0.20359
3	256	8	0.20359
3	512	9	0.20359
3	1024	10	0.20359
1	2	1	0.20169
1	4	2	0.20192
1	8	3	0.20192
1	16	4	0.20192
1	32	5	0.20192
1	64	6	0.20192
1	128	7	0.20192
1	256	8	0.20192
1	512	9	0.20192
1	1024	10	0.20192