Appendix A

TUCANNON: A DDoS Attack Generation Tool

Here, I present a DDoS attack traffic generation tool to generate DDoS attack packets with different characteristics in a controlled testbed environment. My aim is to present a tool which can be used by the researchers to generate DDoS attack traffic with different specifications for the evaluation of their defense system.

A.1 Introduction

In DDoS detection and mitigation research, a major element is the attack traces. These traces are important to understand the characteristics of different types of DDoS attacks as well as to evaluate the performance of a proposed DDoS defense technique under different possible attack scenarios. There are only a few publicly available attack traffic traces such as CAIDA-2007 and DARPA. However, these traces do not represent different possible attack scenarios. For example, in CAIDA-2007 trace all the packets are of same length and of same protocol. Hence, one can try to classify the packets based on these two properties, which is obviously misleading. To generate attack traffic one can use publicly available DDoS attack tools such as TRINOO [22], TFN [22] and LOIC [32]. However, these tools do not have the sophistication to control different features of the attack traffic such as traffic rate, source IP variation and protocol. Hence, as part of my research I have developed a DDoS attack generation tool referred to as TUCANNON which, can generate different types of network and transport layer DDoS attack traffic such as

- TCP SYN flooding attack: A stream of TCP SYN packets towards a specific

victim. The SIPs of the packets either may remain static or may change dynamically.

- TCP ACK flooding attack: A stream of TCP ACK packets towards the victim. The SIPs of the packets either may remain static or may change dynamically.
- TCP Flooding attack: In this type of attack packets TCP flag variables are set randomly. The SIPs of the packets either may remain static or may change dynamically.
- UDP flooding attack: A stream of UDP packets towards the victim. The SIPs of the packets either may remain static or may change dynamically.
- Low rate attack: A stream of attack packets whose volume is comparatively low. The SIPs of the packets either may remain static or may change dynamically.
- High rate attack: A huge volume of attack packets towards a victim IP. The SIPs of the packets either may remain static or may change dynamically.
- Reflection DDoS attack: A huge surge of reply packets from a set of reflector devices. The packets carry the genuine IP addresses of the reflectors.
- Randomly spoofed DDoS attack: A stream of attack packets whose SIPs are randomly chosen from a given range.
- Selected spoofing: A stream of attack packets whose SIPs are spoofed from a set of selected IP addresses.
- Constant rate DDoS attack: A stream of attack packets flowing at constant rate.
- Increasing rate DDoS attack: A stream of attack packets whose volume increases gradually over time.
- Pulse DDoS attack: A stream of attack packets where the packets are generated in bursts separated by some idle time.
- Subgroup DDoS attack: A stream of attack packets where bursts of attack packets are generated by different groups of attack sources.

Table A.1 presents a brief comparison of TUCANNON with other existing tools. In [3, 89], the authors discussed different DDoS attack tools used in the past to perform various DDoS attacks.

A.2 Basics of TUCANNON

A pictorial description of the attack strategy adopted by the tool is shown in Figure A.1. The tool comprises of two components.

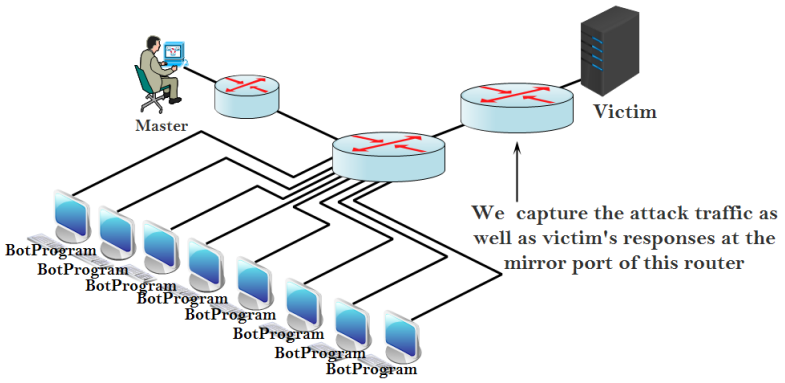


Figure A.1: TUCANNON attack traffic generation strategy

1. The first component is used by the attacker to communicate with the bots. This program uses GUI so that the attacker can easily specify various parameters such as protocol type, attack pattern type, source IP type and packet rate. This program is referred to as *server program*. In figure A.1 the node labeled as master executes the server program. The *server program* is developed using C#. Microsoft windows based machine is required to execute the *server program*.

2. The other component is executed in each bot. This program is responsible for accepting command from server program and launch the attack accordingly. This program is referred to as *client program* program. In figureA.1 the nodes labeled as BotProgram execute the *client program*. The *client program* is written in C language. Raw sockets are used to send attack packets to the victim. Linux based operating system is required to execute the *client program*.

A.2.1 Server Program

Using this program the master communicates with the machines which are configured as bots in the testbed. This program is developed with an user interface through which one can easily specify and control different properties of the attack traffic. Such properties are protocol type (TCP, UDP and ICMP), attack pattern (constant rate attack, increasing rate attack and pulsing attack) and type of source IP (actual IP of the machine or randomly generated valid but spoofed IP address), no of threads (where each thread executes one copy of the slave program inside a single bot machine) and range of ports of the victim to send the traffic. When the master starts, it waits for slaves to connect to it. Figure A.2 is a snapshot of the GUI of the *server program*.

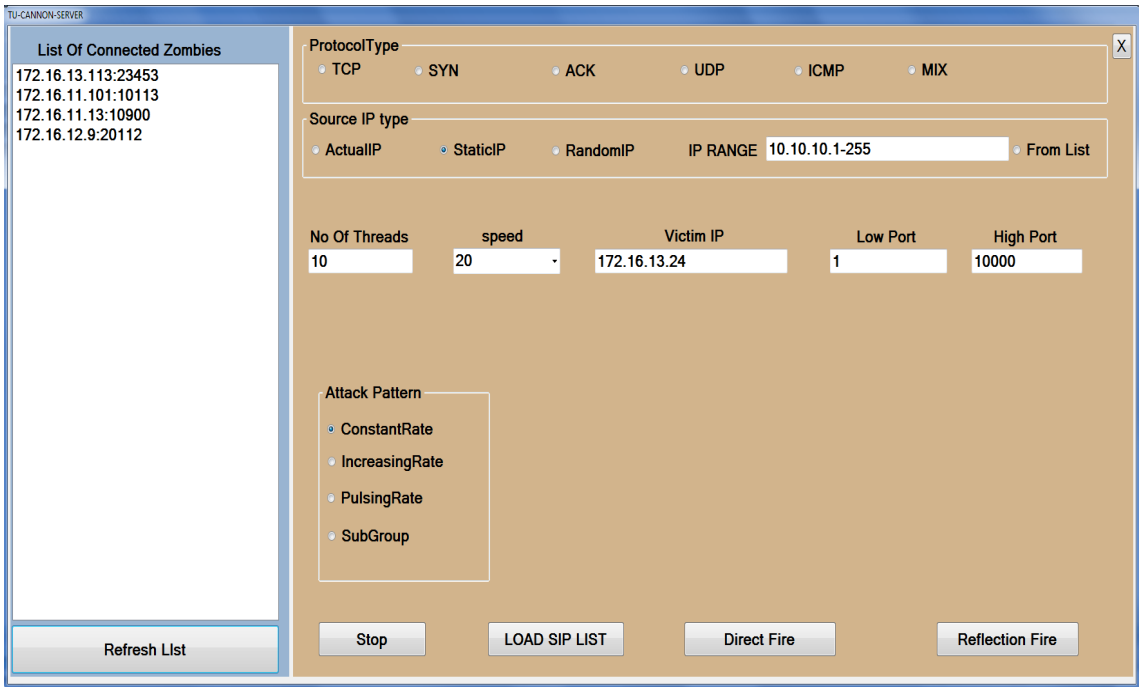


Figure A.2: GUI of TUCANNON server program.

The various components of the interface are discussed below.

1. **List of Zombies:** When the master starts the server program, it waits for the client programs to connect to it. As soon as a client program connects to the server, it's IP address along with the port number it is listening to is shown in the left side panel of the interface. Figure A.2 shows a scenario where there are four bots connected to the master.

2. **Protocol type:** The options under this category allow the master to specify the protocol type of the attack packets. Following are the options available

Table A.1: Comparison of attacking tools

Tool's name	Input	Protocol	Purpose	Effectiveness	Sources
Slowloris	T	HT	DoS	powerful for HTTP attack	www.ha.ckers.org/slowloris
Blackenergy	S	C/U/ IC	DDoS	simple and powerful for DDoS	www.airdemon.net
HOIC	T	HT	DDoS	very effective for DDoS	www.rapidshare.com
Shaft	V	U/C/IC	DDoS	multi-platform, commonly used	
Knight	V	C/U	DDoS	less powerful	www.cert.org
Kaiten	V	U/C	DDoS	Windows based	www.mcafee.com
RefRef	T		DDoS	effective for DDoS	www.hackingalert.net
Hgod	T/p	C/U/IC	DDoS	easy to use	www.flylib.com
LOIC	T/p	C/U/ HT	DDoS	very effective, powerful for flooding attack	www.sourceforge.net
Trinoo	T/p	U	DDoS	multi-plateform, easy to use	www.nanog.org
TFN	T	U/C/IC	DDoS	multi-platform, effective for flooding attacks	www.codeforge.com
TFN2K	T	U/C/IC	DDoS	simple and easy to execute	www.goitworld.com
Stachaldraht	T	C	DDoS	multi-platform, supports more features	www.packetstormsecurity.org
Mstream	T	C	DDoS	multi-platform and more primitive	www.ks.uiuc.edu
Trinity	T	C/U	DDoS	very effective to compromise hosts	www.garykessler.net
TUCANNON	T/p	C/U/ HT	DDoS	Effective for constant rate, increasing rate, pulsing rate and subgroup attack	
Here, T-Target IP, V-Victim IP, S-Server IP, C-TCP, U-UDP, IC-ICMP, F-Input text file, p-Port, HT-HTTP					

in this category.

- **TCP:** Generate TCP attack packets with random TCP flag variable.
- **SYN:** Generate TCP attack packets with the flag variable set to SYN.
- **ACK:** Generate TCP attack packets with the flag variable set to ACK.
- **UDP:** Generate UDP attack packets to different ports of the victim.
- **ICMP:** Generate ICMP echo messages to the victim.
- **MIX:** Generate a mix of all the above types of attack packets to the victim.

3. **Source IP type:** These options are used to control the source IP(SIP) address of the generated attack packets. Following are the different options available.

   - **Actual IP:** This option allows the attack packet to carry the actual IP address of the bot machine. For example, if there are four bots connected

to the master, under this option all the attack packets will contain only these four different SIPs.

- **Static IP:** A single bot can use different threads to generate the attack traffic, where each thread can assume its own IP address (see number of threads option below). If the master selects static IP then each thread will be assigned a randomly selected but fixed SIP. For example, if there are 4 bots and each has 10 threads then under this option the attack traffic will appear as coming from 40 different IP addresses.
- **Random IP:** This option allows each threads to select the SIP address of each attack packet randomly. From the victim's point of view, the attack packets will appear to come from many different IP addresses.
- **IP range:** This option allows the master to specify the range of IP addresses which are to be used as the SIP of the attack packets by the attacking threads. For example, In Figure A.2, the IP range is specified as 10.10.10.1-255, hence the generated attack packets may carry any SIP from 10.10.10.1 to 10.10.10.255.
- **From list:** Instead of an IP range, the master can also specify a list of IP addresses from where the attacking threads can chose the SIP addresses of the attack packets. This option is useful to perform a reflection DDoS attack, where the master can specify the IP addresses of the reflector devices in the list.

4. **No of threads:** As mentioned above, a single bot can use different threads to generate the attack traffic. This option allows the master to specify the number of threads to be used by a single bot machine.

5. **Speed:** Through this option the master can specify the approximate packet rate of a single thread. As shows in Figure A.3, the master is allowed to select the speed from a drop down list. The packet rate seen by the victim will be approximately the product of the number of bots connected to the master and the number of threads used by each bot machine. For example, if there are 4 connected bots and each bot uses 10 threads to generate 20 packets per second then, the final attack traffic received by the victim will be approximately $4 \times 10 \times 20 = 800$ packets/sec.

6. **Victim IP:** This input field is used by the master to specify the IP address of the victim machine.

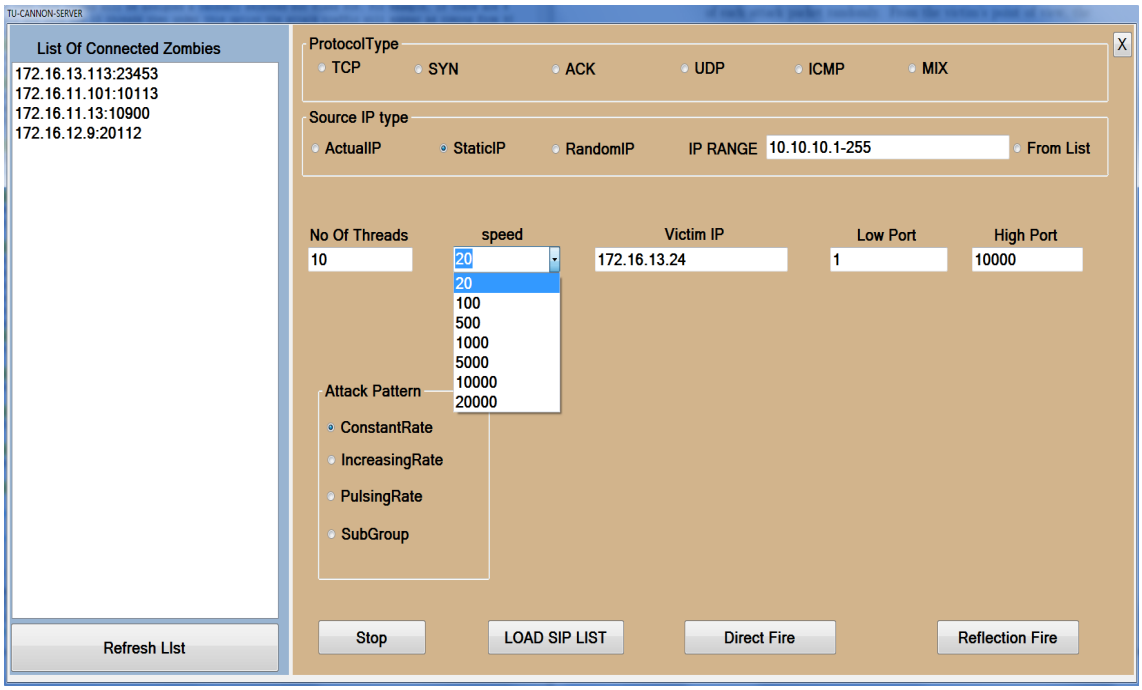


Figure A.3: Selecting packet rate of an attack thread

7. **Low port and high port:** These two options specify the range of destination port numbers of the generated attack packets.

8. **Attack pattern:** The options available under this category allows the master to generate different attack patterns such as constant rate attack, increasing rate attack , pulsing attack and subgroup attack. Following are the description of each attack pattern.

    - **Constant rate:** This option will cause each thread in a bot machine to generate the attack packets towards the victim instantly and continuously.
    - **Increasing rate:** Under this option each bot machine gradually increases the number of attack threads rather than activating all of them at once.
    - **Pulsing rate:** This option allows the master to generate pulses of attack packets. The master can specify the duration of the attack pulses as shown in Figure A.4. The on duration represents the length of an attack burst in time. The off duration represents the intermediate time interval between two adjacent attack bursts. The SIP addresses of the attack packets will be populated based on the option selected for source IP type.

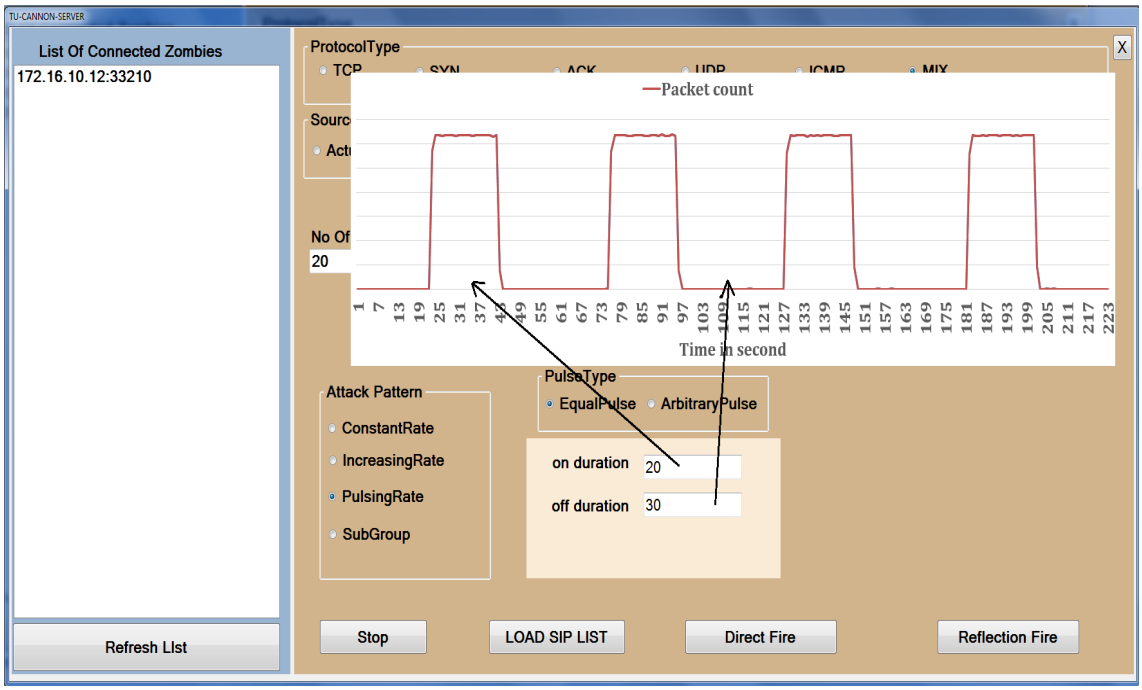


Figure A.4: Selecting the on/off period of a pulse attack

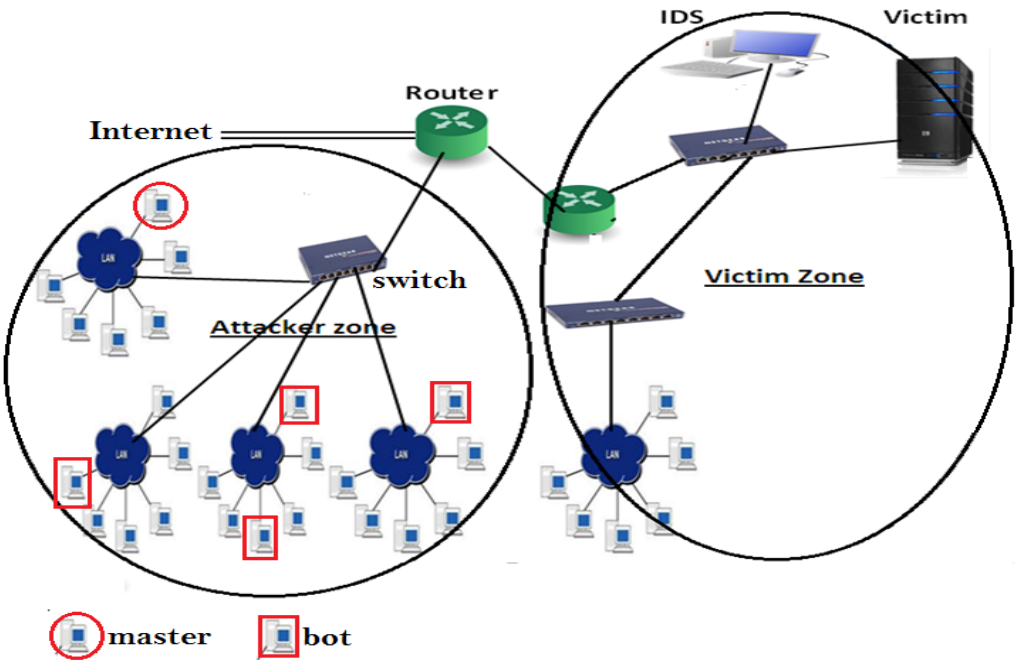


Figure A.5: The testbed setup for TUCANNON experiments experiments.

- **Sub group attack:** The subgroup attack option is similar to pulse attack with one difference. In subgroup attack a different SIP is used for each attack burst by a thread. Thus, at the victim end each attack burst will appear to come from a different group of attack sources.

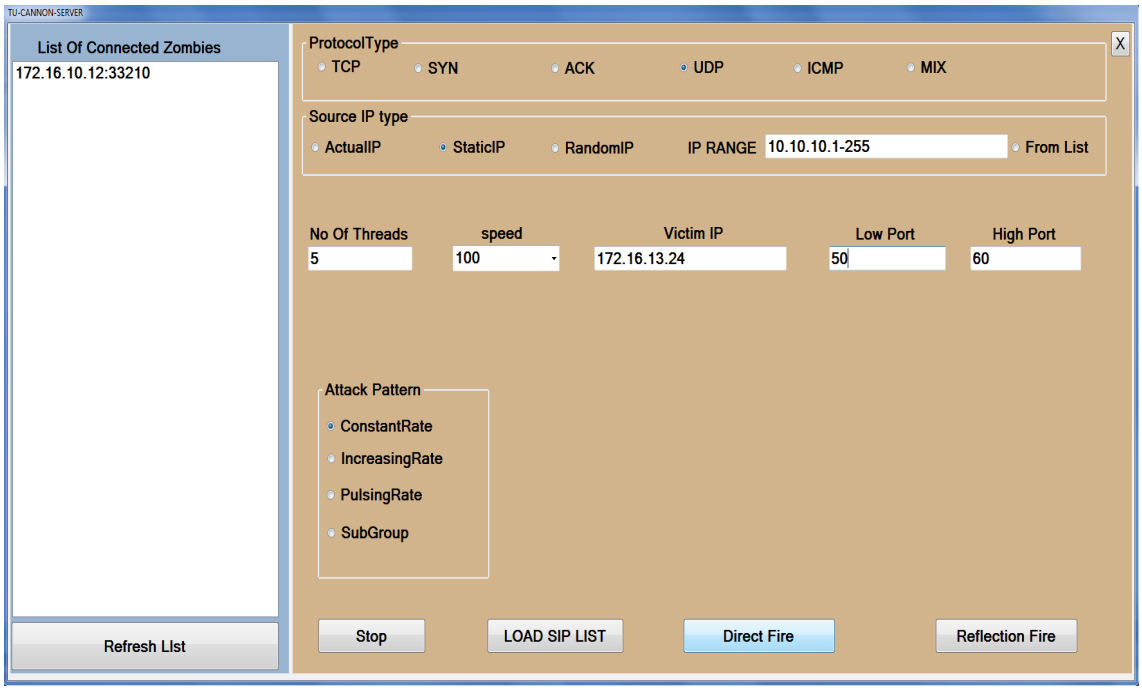


Figure A.6: Parameter specification for attack scenario 1

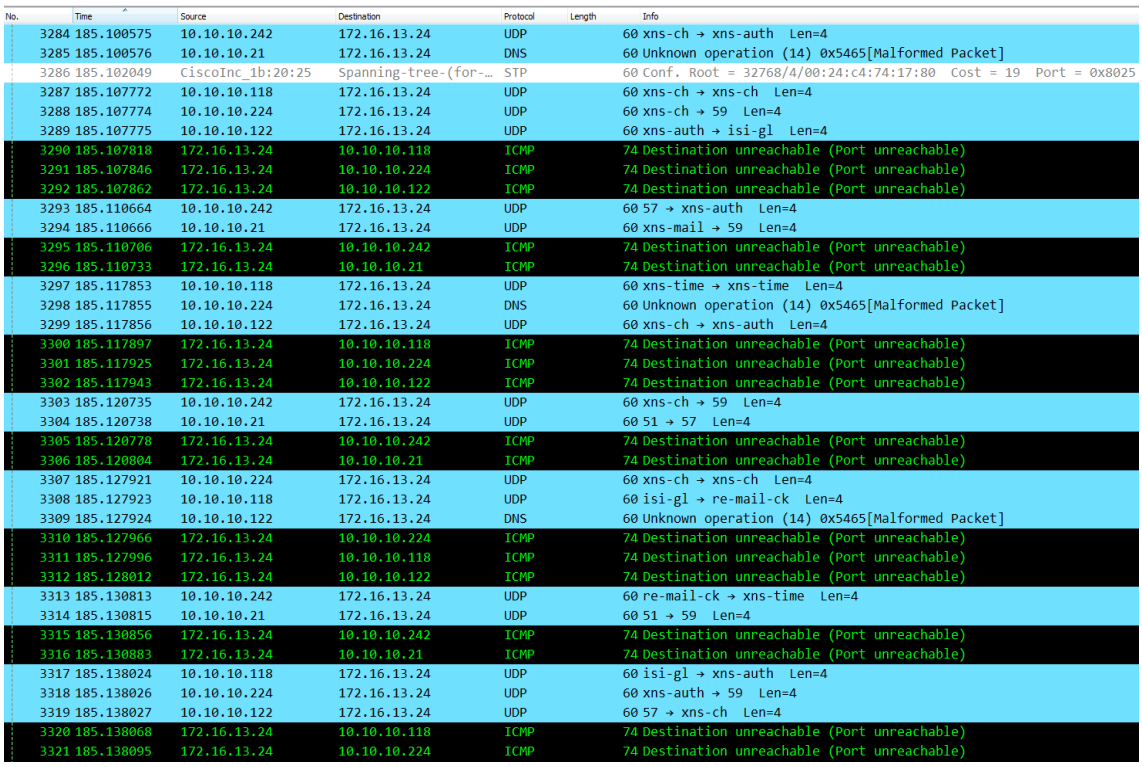
9. **Load SIP Llist:** This option allows the master to load a list of IP addresses to the bot machines. The bots can be instructed to take the SIP of the attack packets from this list by using *from list* option under *source IP type* category, as mentioned above.

10. **Direct fire:** When the attacker clicks on this button, attack command along with the specified input is sent to all clients connected to the server. As a result, the victim specified by the *victim ip* field starts receiving attack packets from the bot machines.

11. **Reflection fire:** This button allows the master to carry out a reflection attack. A reflection attack is different from a direct attack as follows.

    - The source IP field of the attack packets are filled with the IP address specified in the *victim IP* option in the UI.
    - The attack packets are sent to the IP addresses specified in the *IP range* or *from list* option.

    Thus, under this attack type, many different IP addresses receive packets (such as ICMP echo request) containing the victim's IP as the SIP of the packets. As a result, all of them replies to the victim with appropriate reply messages (such as ICMP reply) causing a huge surge of packets towards the victim.



No.	Time	Source	Destination	Protocol	Length	Info
3284	185.100575	10.10.10.242	172.16.13.24	UDP	60	xns-ch → xns-auth Len=4
3285	185.100576	10.10.10.21	172.16.13.24	DNS	60	Unknown operation (14) 0x5465[Malformed Packet]
3286	185.102049	CiscoInc_1b:20:25	Spanning-tree-(for-…	STP	60	Conf. Root = 32768/4/00:24:c4:74:17:80 Cost = 19 Port = 0x8025
3287	185.107772	10.10.10.118	172.16.13.24	UDP	60	xns-ch → xns-ch Len=4
3288	185.107774	10.10.10.224	172.16.13.24	UDP	60	xns-ch → 59 Len=4
3289	185.107775	10.10.10.122	172.16.13.24	UDP	60	xns-auth → isi-gl Len=4
3290	185.107818	172.16.13.24	10.10.10.118	ICMP	74	Destination unreachable (Port unreachable)
3291	185.107846	172.16.13.24	10.10.10.224	ICMP	74	Destination unreachable (Port unreachable)
3292	185.107862	172.16.13.24	10.10.10.122	ICMP	74	Destination unreachable (Port unreachable)
3293	185.110664	10.10.10.242	172.16.13.24	UDP	60	57 → xns-auth Len=4
3294	185.110666	10.10.10.21	172.16.13.24	UDP	60	xns-mail → 59 Len=4
3295	185.110706	172.16.13.24	10.10.10.242	ICMP	74	Destination unreachable (Port unreachable)
3296	185.110733	172.16.13.24	10.10.10.21	ICMP	74	Destination unreachable (Port unreachable)
3297	185.117853	10.10.10.118	172.16.13.24	UDP	60	xns-time → xns-time Len=4
3298	185.117855	10.10.10.224	172.16.13.24	DNS	60	Unknown operation (14) 0x5465[Malformed Packet]
3299	185.117856	10.10.10.122	172.16.13.24	UDP	60	xns-ch → xns-auth Len=4
3300	185.117897	172.16.13.24	10.10.10.118	ICMP	74	Destination unreachable (Port unreachable)
3301	185.117925	172.16.13.24	10.10.10.224	ICMP	74	Destination unreachable (Port unreachable)
3302	185.117943	172.16.13.24	10.10.10.122	ICMP	74	Destination unreachable (Port unreachable)
3303	185.120735	10.10.10.242	172.16.13.24	UDP	60	xns-ch → 59 Len=4
3304	185.120738	10.10.10.21	172.16.13.24	UDP	60	51 → 57 Len=4
3305	185.120778	172.16.13.24	10.10.10.242	ICMP	74	Destination unreachable (Port unreachable)
3306	185.120804	172.16.13.24	10.10.10.21	ICMP	74	Destination unreachable (Port unreachable)
3307	185.127921	10.10.10.224	172.16.13.24	UDP	60	xns-ch → xns-ch Len=4
3308	185.127923	10.10.10.118	172.16.13.24	UDP	60	isi-gl → re-mail-ck Len=4
3309	185.127924	10.10.10.122	172.16.13.24	DNS	60	Unknown operation (14) 0x5465[Malformed Packet]
3310	185.127966	172.16.13.24	10.10.10.224	ICMP	74	Destination unreachable (Port unreachable)
3311	185.127996	172.16.13.24	10.10.10.118	ICMP	74	Destination unreachable (Port unreachable)
3312	185.128012	172.16.13.24	10.10.10.122	ICMP	74	Destination unreachable (Port unreachable)
3313	185.130813	10.10.10.242	172.16.13.24	UDP	60	re-mail-ck → xns-time Len=4
3314	185.130815	10.10.10.21	172.16.13.24	UDP	60	51 → 59 Len=4
3315	185.130856	172.16.13.24	10.10.10.242	ICMP	74	Destination unreachable (Port unreachable)
3316	185.130883	172.16.13.24	10.10.10.21	ICMP	74	Destination unreachable (Port unreachable)
3317	185.138024	10.10.10.118	172.16.13.24	UDP	60	isi-gl → xns-auth Len=4
3318	185.138026	10.10.10.224	172.16.13.24	UDP	60	xns-auth → 59 Len=4
3319	185.138027	10.10.10.122	172.16.13.24	UDP	60	57 → xns-ch Len=4
3320	185.138068	172.16.13.24	10.10.10.118	ICMP	74	Destination unreachable (Port unreachable)
3321	185.138095	172.16.13.24	10.10.10.224	ICMP	74	Destination unreachable (Port unreachable)

Figure A.7: Wireshirk snapshot of the attack packets in attack scenario 1

12. **Stop:** The attacker can stop the attack by clicking on this button.

A.2.2 Client Program

This program is responsible for actually sending the attack traffic as specified by the command sent from the master. When the client program starts, it connects to the server whose IP is specified as input to the client program. After connecting to the server it waits for command from the server.

A.3 Demonstration of TUCANNON

In this section, I demonstrate how TUCANNON can be used to generate DDoS attack traffic with different specifications. A pictorial representation of the testbed used to generate the attack traffic for the experiments is shown in figure A.5. The testbed consists of two subnetworks.

1. The first subnetwork (the left side oval) is used to generate the attack traffic. One of the machine in this subnetwork is used to run the *server program* of TUCANNON. The *client program* is executed from multiple other machines

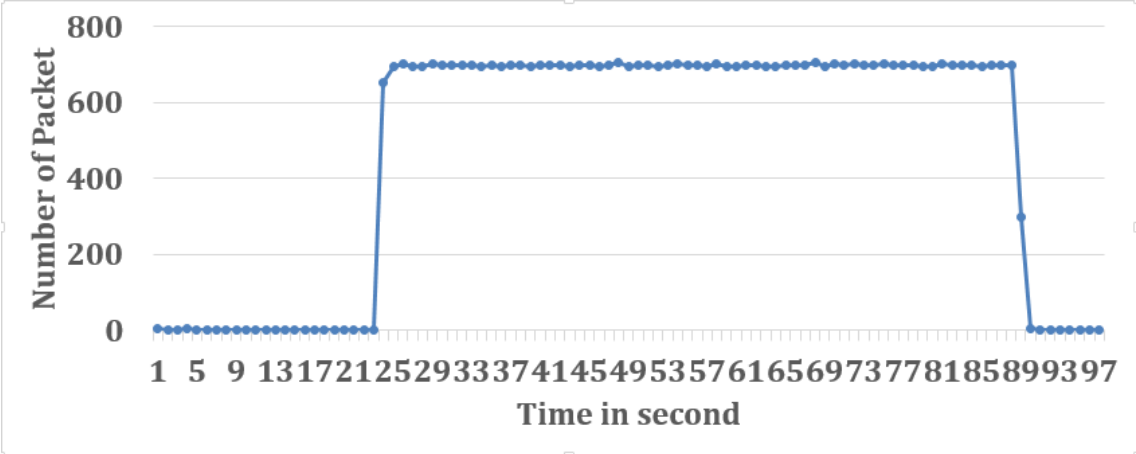


Figure A.8: Attack scenario 1 packet rate

from this subnetwork. One possible arrangement is depicted in Figure A.5, by marking the *server program* as master and the *client program*s as bot.

2. The second subnetwork (the right side oval) contains the victim server, labeled as victim in Figure A.5, and the packet capturing point.

For demonstration I use a single bot which is connected to the master. Since, each bot executes the same *client program*, traffic from multiple bots will also follow the same pattern. Following attack scenarios are used to demonstrate the working of TUCANNON.

A.3.1 Attack Scenario 1

In this experiment, TUCANNON is used to mimic a UDP flooding attack coming from 5 different attack sources at around 100 packets/second from each attack source. The attack packets are generated at a constant rate. The SIP address of the attack packets are spoofed from 10.10.10.∗ subnetwork. Also, the destination port numbers of the attack packets are chosen to be in between 50 and 60. To generate the attack, the master can specify the parameters as shown in Figure A.6. A Wireshirk [90] snapshot of the captured packets at the monitoring point is shown in Figure A.7. In the figure each row represents a single packet. The columns represent different attributes of the captured packets such as time stamp, source IP, destination IP, protocol, packet length and port information. The incoming attack packets and corresponding out going response packets are shown in cyan and black color respectively. From the *source IP* column, we can see that the attack packets are appeared to come from 5 different sources whose IP addresses are in 10.10.10.∗

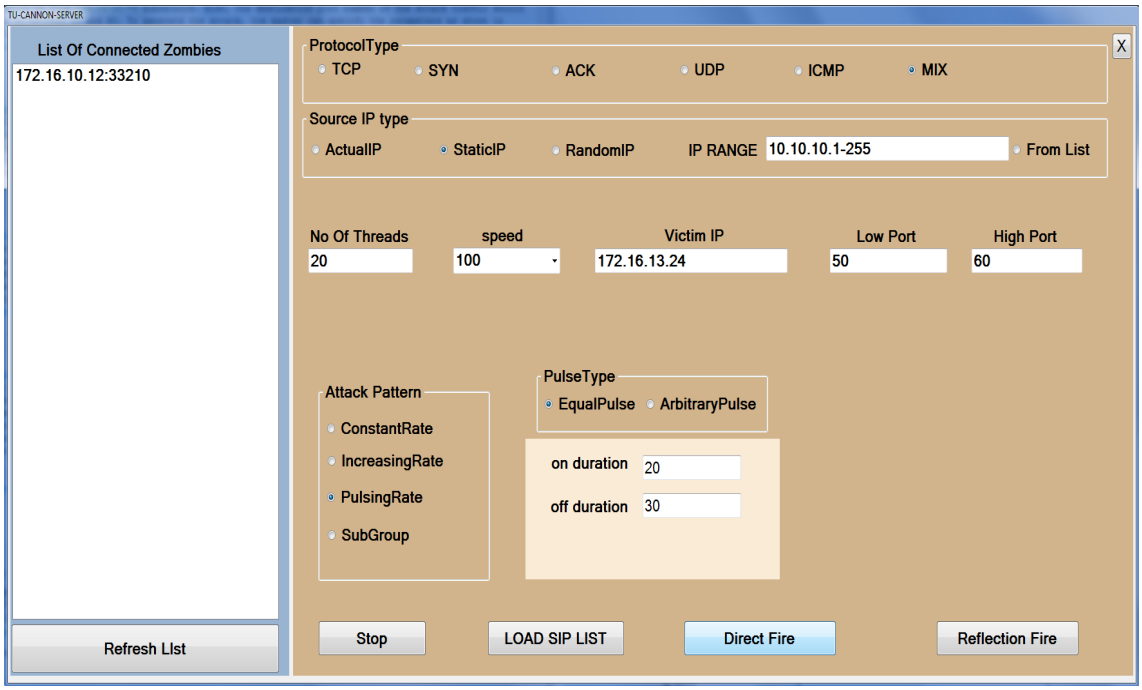


Figure A.9: Parameter specification for attack scenario 2

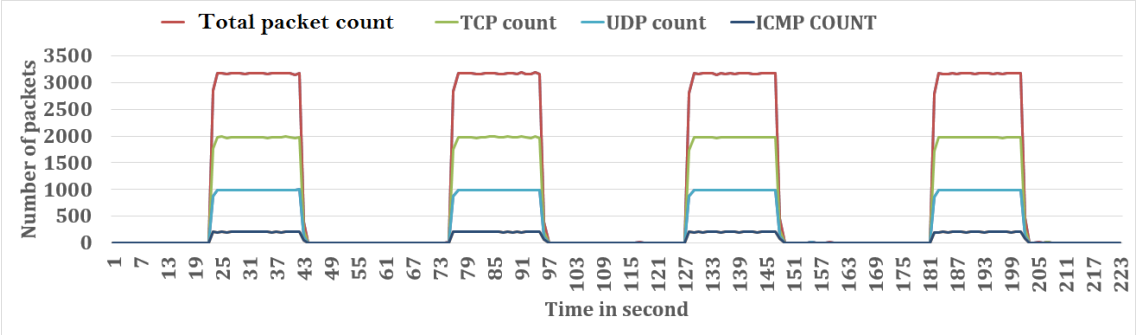


Figure A.10: Packet rate decomposition for different protocol in attack scenario 2

range. Similarly, the destination IP, protocol and port info columns also confirm the configuration provided in the UI shown in Figure A.6. Figure A.8 shows the packet rate of the aggregated attack traffic at the monitoring point.

A.3.2 Attack Scenario 2

In this experiment, TUCANNON is used to mimic a MIX flooding attack (i.e. attack packets are of different protocol type) coming from 20 different attack sources at around 100 packets/second from each attack source. The attack pattern is selected as pulsing attack, where the attack sources send burst of attack packets of duration 20 seconds after each 30 seconds. The SIP address of the attack packets are selected from 10.10.10.$*$ subnetwork. Also, the destination port number of the

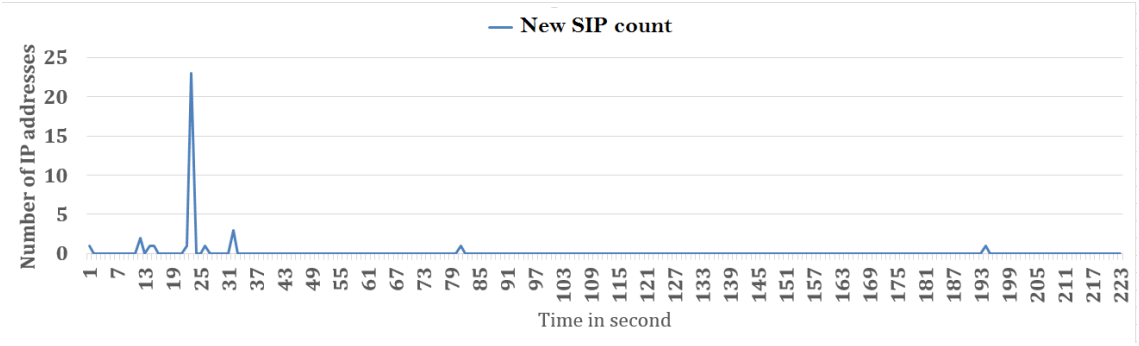


Figure A.11: New SIP rate in attack scenario 2

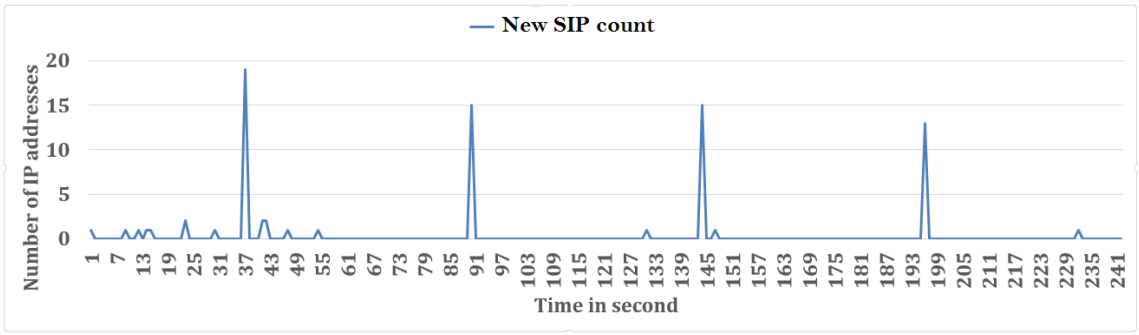


Figure A.12: New SIP rate in attack scenario 3

attack packets are chosen to be in between 50 and 60. To generate the attack, the master can specify the parameters as shown in Figure A.9. Figure A.10 shows the packet rate decomposition of the aggregated attack traffic at the monitoring point based on their protocol. We can see that the attack trace contains TCP, UDP and ICMP packets at different rate. Figure A.11 shows the number of new SIP seen at each interval of one second. The spikes represents the arrival of one or more new SIPs. From Figure A.11 we can see that even if the attack trace contains multiple bursts, all the bursts are generated using the same set of attack sources represented by the spike in between 11^{th} and 25^{th} interval.

A.3.3 Attack Scenario 3

This experiment is same with attack scenario 2 except, in this experiment a subgroup attack is generated instead of a pulse attack. The attack pattern is similar to that of attack scenario 2 as shown in Figure A.10. However, we can see a difference in the new SIP rate pattern as shown in Figure A.12. We can see that instead of generating the attack bursts from the same set of sources, in subgroup attack different bursts are generated by different group of sources represented by the spikes at 37^{th}, 91^{th}, 145^{th} and 193^{th} intervals in Figure A.12.

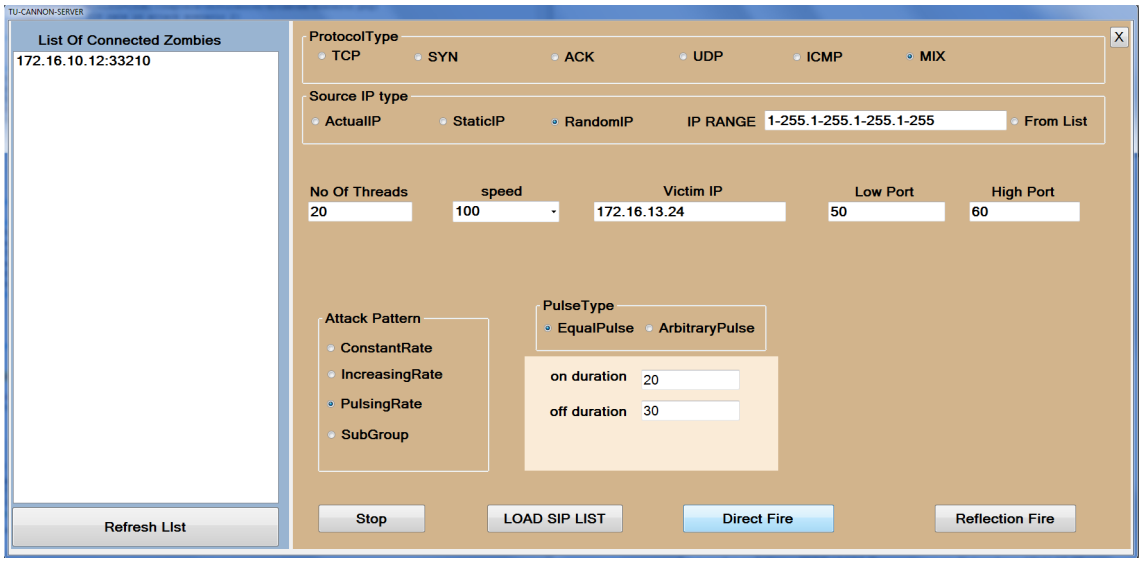


Figure A.13: Parameter specification for attack scenario 4

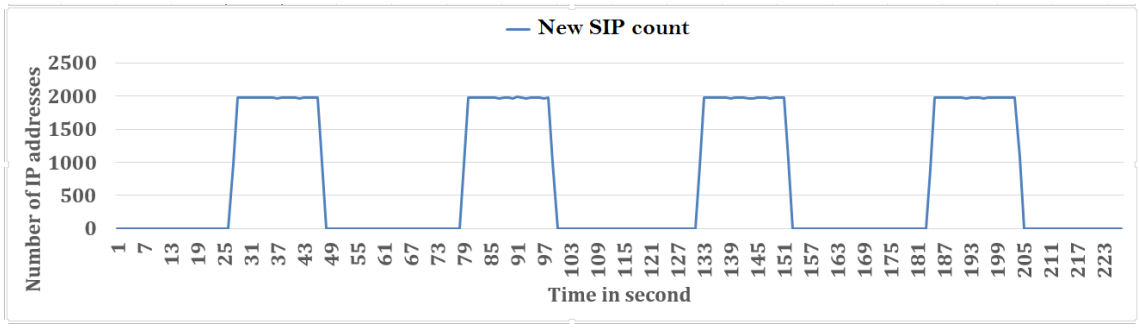


Figure A.14: New SIP rate in attack scenario 4

A.3.4 Attack Scenario 4

This experiment is same with attack scenario 2 except, in this experiment the SIP of each of the attack packets are chosen randomly from $*.*.*.*$. The parameter specification is shown in Figure A.13 .The attack pattern is similar to that of attack scenario 2 as shown in Figure A.10. Figure A.14 shows the new SIP rate of the attack trace. We can see that at every interval thousands of new SIP addresses are detected at the monitoring point. Even if only 20 threads are used to generate the attack packets, due to the *random IP* option selected in *source IP type* category all generated packets are spoofed with random SIPs.

A.4 Discussion

Here, I discussed TUCANNON, a DDoS attack traffic generation tool. TUCANNON allows a researcher to easily setup a botnet of multiple machines spread over a controlled testbed subnetwork. TUCANNON comes with an UI which allows the user of the tool to control the botnet from a single point. The UI allows the user to specify different parameters of the generated attack packets. I demonstrated the working of TUCANNON using different attack scenarios.

TUCANNON is developed to generate DDoS attack packets with minimal effort by a researcher. The generated packets however, does not follow normal Internet communication pattern. Hence, the packets generated by the tool should not be used to model normal Internet traffic.