International Journal of Computational Intelligence Research
ISSN 0973-1873 Volume 12, Number 1 (2016), pp. 83-94

http://www.ripublication.com

Identifying Bot Flooding Attack using NTP

Ruqayya Siddiqui[1] and Anchit Bijalwan[2]

Department of computer science
Uttaranchal University Dehradun, Uttarakhand, India
E-mail: [1]riyyu.reema@gmail.com, [2]anchit.bijalwan@gmail.com

Abstract:

The Internet is so far one of the most innovative discoveries ever found. The Internet has made it possible for us to do lots of things or everything like ecommerce, communications, entertainment, data storage, and so much more. But also here are some disadvantages of Internet are: Leakage of private information, spam mail, trojan or malware attacks. These are weaknesses of Internet; the interesting activities are all are occurred from Internet and attacker easily attacks on user's systems or devices because of the protocol stack.

Network forensics is a sub part of digital forensics and also it is controlled under digital forensics. Main working of network forensics focused on collection of digital evidence and analysis of problems or packets which comes through intruder for analytical purposes.

Flooding attack simple as DoS attack, in UDP-flooding attack; attacker send several UDP datagram of unlike sizes at same time. It is similar to a chain association for systems to hide identity. For forensic inquiry in this paper we introduce a new protocol Net Token Protocol (NTP), which is helpful in network based activity. In this protocol token processing is beneficial as a system chain connection and the protocol are mainly protect to those users whose capable to returning tokens which is useful for connection of information.

1. Introduction:

We develop a new protocol to support forensic analysis of spiteful network-based activity; First understanding of botnet is significant. The word botnet is completing awake of two words, bot and net. Bot is short in favor of robot, a name we sometimes provide to a computer that is impure by malicious software. Net comes as of network, a group of systems that are connected together. People who inscribe and operate malware cannot physically log onto every computer they have infected, instead they

use botnets to control a large number of impure systems, and do it involuntarily. A botnet is a network of tainted computers, where the network is used through the malware to spread.

A UDP (User Datagram Protocol) is a transport layer protocol distinct for exploit with the IP network layer protocol. UDP flood is a network flood and still a standout amongst the most widely recognized floods today. The attacker sends UDP packets, in general huge ones, to single destination or to arbitrary ports. In most cases the attackers spoof the Source IP which is easy to do because the UDP protocol is "connectionless" and does not have any type of handshake mechanism or session. This advance causes denial of service (DoS) attack. It is more risky, if we disturb or try to change in flood. In other case attackers use a chain association through many systems to cover identity, for mitigation of this attack we propose a new protocol.

Our work related to a proposed protocol, the Net Token Protocol (NTP), it upgrade the ident communications by distribution of recursive requests to previous devices on the connection chain. Main purpose of protocol is protection of user's and privacy hiding by returning a token that is a sub code of connection data. At the end here decision is on system administrator for information sharing of token to other systems.

The Malicious node or attacker system generates multiple UDP floods; they have no any restriction for across the network and floods easily enter in client's systems. Primary expectation of a UDP flood is to saturate the Internet connection. Another effect of this attack is on the system and security components while in transit to the objective server, and most commonly the firewalls. Firewalls open a circumstance for each UDP packet and will be overpowered by the UDP flood connections quick and attack can be performed very fast, in particular addressing the stepping-stone setting in which an attacker uses chain of associations (figure 1) through many hosts to hide his identity.

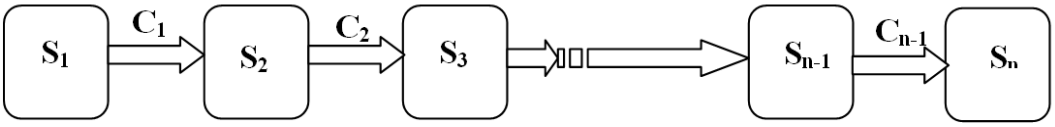

Figure 1. Connection chain between system $S_1$to S_n

2. Related Work:

Yuan Tao et al. [1] proposed DDoS attack detection scheme for local area networks. Flow entropy is employed on the LAN routers to supervise the traffic and to raise the potential flooding alarms. An information distance is used differentiate between false alarms and DDoS attacks. The Mathematical models are implemented for the proposed detection schemes. During the experimentations, it has been observed that the proposed schemed is very effective to detect the DDoS attacks. Moustis et al. [2] analyzed DDoS attacks that require only a small number of bots to make a web server unavailable. The bots are simulated by using both Windows and Linux based systems infected with Slowloris (HTTP syn-flooder), targeting to a web server. Several

security controls are also applied to test the effectiveness of proposed method against such attacks. In simulations, it has been observed that a combination of carefully selected anti-DDoS controls can reduce the exposure of flooding attack. Hussain et al. [3] showed the effect of UDP flooding on the performance of the number of queuing algorithms like Droptail (DT), Random Early Discard (RED), Deficit Round Robin (DRR), Fair Queue (FQ) and Stochastic Fair Queue (SFQ) is measured. During the experimentation, it has been observed that SFQ performs better for UDP traffic as compared to the other schemes. In Bardas et al. [4], authors present the investigation of proportional-packet rate assumption. The classification of UDP traffic is done, the objective is to detect malicious addresses that cause UDP flooding attack. In the experiments the dataset is created by taking data from ISPs, universities, financial institutions, etc. A prototype classifier is implemented and a method is also discussed, how it can be used to prevent the UDP flooding attacks. Silva et al. [5] reviewed on botnet problem. Author summarized the previous work related to botnet attacks, the problems and some solutions to those problems are also discussed. The open prominent and persistent research problems of botnet are also discussed**.** Mansfield et al. [6], a discussion on botnet and whitehats is done. There is a continuous arms race between botnet operators and the whitehats (researchers), anti-malware organization and law enforcement organizations. The most visible action of this conflict is the malware, but there is a less obvious struggle going on to control the infrastructure, supports the unauthorized actions of botnet operators. By the application of malware, the botnet operators can build and manage their infrastructures more effectively, as seen in the past few years. In Rui et al. [7], an artificial immune detection based defense system against UDP flooding attack is proposed. The r-bits matching rule is introduced with eigenvalue matching scheme. The all non self modes are detected by the application of eigenvalue filter windows. In simulation, it has been observed that the proposed defense system detects the fake IP addresses from UDP flooding successfully. In Argyraki et al. [8], proposed an Internet traffic filtering (AITF), a network-layer defense technique against bandwidth consuming flooding attacks. The proposed scheme enables a receiver to contact to the misbehaving source and ask him to stop the flooding traffic. The each flooding source that has been asked to stop is policed by its own Internet service provider (proposed method examines DNS logs from the destination to the source, in order to detect the bots. A technique is also proposed to distinguish between spoofing from non-spoofing attacks**.** Park et al. [9] proposed an SNMP- based lightweight and fast detection technique for traffic flooding attacks. It minimizes the processing and network overhead of the intrusion detection system, the detection time, and provides high detection rate.ISP). AITF protects the network against the flooding and also reduced the bandwidth consumption. It is also shown that, two networks deployed with AITF scheme can maintain their connectivity to each other in the presence of flooding. Takemori et al. [10] proposed an IP tracking scheme against bot attacks using the DNS logs. Safaa et al. [11] proposed a defense mechanism against SYN flooding is proposed. It makes the use of spoofed IP addresses associated with edge routers to determine whether the incoming SYN- ACK segment is valid or not. A matching table of the outgoing SYNs and incoming SYN- ACKs are maintained. If the incoming SYN- ACK segment is

invalid, the edge router resets the connection at the victim host, freeing up an entry in the victim's backlog queue, and enables it to accept other legitimate incoming connection requests (RQ). The performance evaluation of the proposed technique is also done. T. Hurth et al. [12] proposed a method for benchmark and derive the consequences of the MFV hypothesis for $\Delta F=1$ flavour observables based on the latest LHCb data. Anil Kurmus et al. [13] explore an alternative, automated and effective way of reducing the attack surface in commodity operating system kernels, which we call trimming. Vural et al. [15] proposed botnet identity concealment techniques. In order to detects botnet computational intelligence techniques are proposed. A simulate for network anomaly detection is done. Anchit et al. [16] proposed a technique for the forensics of Random-UDP flooding attack. They tried to get as close as possible to the source of such attacks. The proposed technique is capable to identify the source of Random-UDP flooding bot attack.

3. Proposed algorithm and protocol:

3.1 Base algorithm for communication of Client/Server and Malicious node

STEP-1. Client tries to communicate with server using web-browser.
STEP-2. Send a HTTP request to web-server.
STEP-3. At server, malicious node extracts client's data and starts flooding to that client.
STEP-4. Attackers use a chain of connections
STEP-5. Attacker response to client and flooding packet come to the client's system like a response.
STEP-6. Proposed protocol NTP works with OS (Linux, OpenBSD).
STEP-7. Comprehensive Benchmark set and works under Phoronix Test Suite.
STEP-8. Phoronix Test Suite performed and tests all process in user's system.
STEP-9. Now filtering with the help of tool and specify those flood packets.
STEP-10. This method is useful for detect source IP of flooding.

3.2 Net Token Protocol

NTP is a proposed protocol which provides some additional functionality from *ident*. Easily it can be work with any system without modification of any other protocols, network topology, or core part of OS. NTP also run in parallel and network connection chain analysis tools, some of the functionalities are follows:

- **Goal:**
  - The client saves additional data, in addition to just the user name.
  - The client traces the user's path of previous hosts.
  - Should allow a system that is not on the connection chain to make requests.
- **Design:** Proposed protocol build under *ident* protocol with multiple request messages to provide more options and multiple request type, 4 main routine of design are follows:
  - **ID:** Same as original *ident* protocol.
  - **ID_R:** It identifies cycle in recursion.

- **SV:** Daemon saves user's name and other data.
- **SV_R:** Save details with recursion property.

- **Saving:** With the help of SV and SV_R request to the user some other details are useful:
  - Process identifier (PID)
  - Parent PID
  - Effective user id
  - Process timing (from starting)
  - Address of request's machine.
  - Address and port of remote end of socket
  - Request type (i.e. SV_R)
  - OS, Version, Kernel.
- **Recursion:** ID_R and SV_R here R refers to request types, it allow tokens to be generates new recursive path of systems**.**
- **Security:** NTP also performs in multiple systems (S_{i-1}, S_i, S_{i+1}), using for connection chain problems and also useful for mitigation of attacks. It is secure protocol in comparison to *ident* protocol.
- Return random tokens.
- Opt-in to releasing their user name.
- Return "UNKNOWN-ERROR".
- For save it select state data.
- Confine the quantity of dynamic lookups to constrain the measure of processing the daemon does.

Using these steps of algorithm we are working on four different methodologies:

1 View normal flow of UDP datagram with DoS attack using Random-UDP flooding.
2 NTP protocol.
3 Performance of Request/Response between user's system and malicious node.
4 System performance for Connect Random-UDP to Forensics.

4 Implementation:

4.1 View flow of UDP datagram

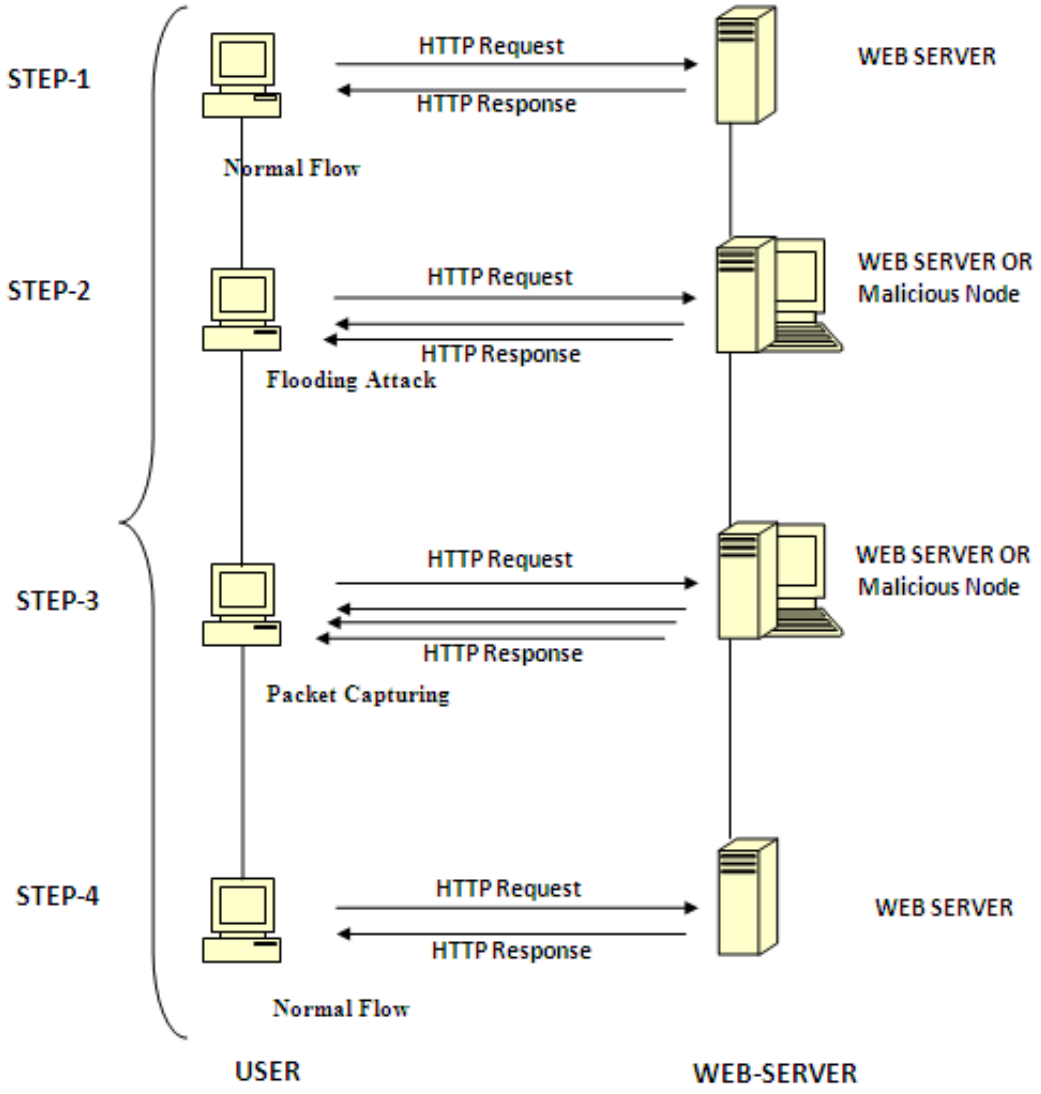

Figure 1. Data flow between User and Server or malicious node

4.2 Working of NTP Protocol

A prototype of the NTP protocol was performed by adapting an open source ident daemon, *oidentd*. It works on user's system and performs request/response in both TCP/UDP. The NTP daemon tolerable several run-time decisions:

Users are ready to send a random token through system, when users are premised to opt-in to his name (user name) being free then next step is name creation of file ˜/.ident, here users enclose evidence of systems so as to their user's name must be sent to.

Here we are using OpenBSD for implementation. It is a better way to implement because it fetch directly from kernel memory. The process state data was determined in OpenBSD by the Kernel VM library utilities and in Linux used the procfs (proc-file system) and the main template was built on OpenBSD and Debian Linux 2.4.
After receiving ID type request, the daemon make a decision for UID and confirm it is from kernel memory and also demonstration same as a vital ident daemon, here location of file in Linux is /proc/net/tcp. Also process identification done by daemon that has the socket and stores state data about it. Now Daemon check and analysis data from parent process then 'walks' up the process tree through analyze and do this procedure again and again awaiting the process with process ID, PID 0 is achieved. 'Walk' period is significant for each socket identification because this time remote end of incoming socket received messages by recursive request.
Process tree may not be significant for performing 'walk' up when tracing malware users. Here it's an example of attacker's command - **Si: # nc -l -p 8888 | nc <Si+1> 8889**
Here *netcat* is helpful to reorganization on port 8888 of system Si and other data like pipe data received through other *netcat* process which sends the data with increment of port means 8889 on system Si+1. Now it's confirmed no other sockets come across after process if connects to Si+1. So if Si+1 proceed request SV_R means here no any recursive requests will be sent. Si-1 determined if pipe resolved and identified at the other end.

4.3 Request/Response Performance

Program worked as it's processing and generated a sub-program (daemon) so as to it is used for implementation of NTP protocol.
Performance completed by many processes in a single operating system with all sub-programs (daemon), for multiple processes the addition of 100 processes. For example here we are taking 6 processes and its new files, Daemon searches all file descriptor to solve its bandwidth means its pipe, so here 600 new files descriptors for 6 reprocesses. And if we compare platforms or operating systems then we analysis Linux and OpenBSD are most useful in this NTP protocol. In table 1 or 2 we are showing ID, SV, SV with file and SV with 80 proc for both platforms and its processing time for both at all levels.

Table 1. Average lookup time for different processes

Platform	ID	SV	SV with file	SV with 80 proc
Linux	0.413 mS	4.318 mS	7.843 mS	218.572 mS
OpenBSD	0.702 mS	2.123 mS	7.271 mS	31.512 mS

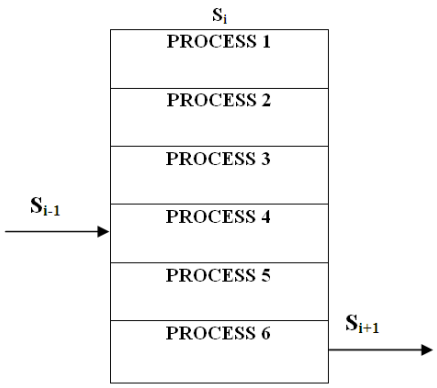

Figure 1. Process tree with 6 distinctive processes

4.4 System Performance

For determining the impact of daemon on a system we are using *Phoronix Test Suite*. The *Comprehensive Benchmark* was executed through this tool and timed exclusive of the daemon successively to resolve the base time. In our example base time are 8 because we are starting and concluding processes from this time later which examine 8 to 5500, meaning that total time is 5492 for both platforms. Request rate are different from base time which shown in table 2. Here for a 6 process tree output printed to a file which relate to SV type request.

For a resultant value, we analyze all computer systems in Uttaranchal University and basically we focused on students computers. Here all computers run under Uttaranchal University Computing Center/Administrator (authority.cc.uttaranchal.edu), and all students are registered on it. We were calculated average number of logins per minute from students computers, over a six hour period, there were 2167 logins, or almost six per minute. Here extreme case checkup is impotent, then we found upper bound case means every user logs into another system after logging into expert in this case; this value is used as an upper bound for the number of request a system may receive a minute.

Table 1. System performance with request rate

Platform	**Request rate per minute**							
	8	**15**	**30**	**150**	**600**	**2500**	**3500**	**5500**
Linux	0.15%	0.32%	0.45%	0.88%	1.25%	22.80%	30.24%	48.96%
OpenBSD	0.2%	0.12%	0.19%	0.23%	0.90%	7.52%	11.05%	19.25%

Daemon perform and handle complex process structure, here we are showing a batch of processes (figure 4) in a tree format which resolves 10 unique processes and perform with multiple systems (S_i), figure 5 shows these values in a graph. Sometime these processes traced by malicious node through Internet socket.

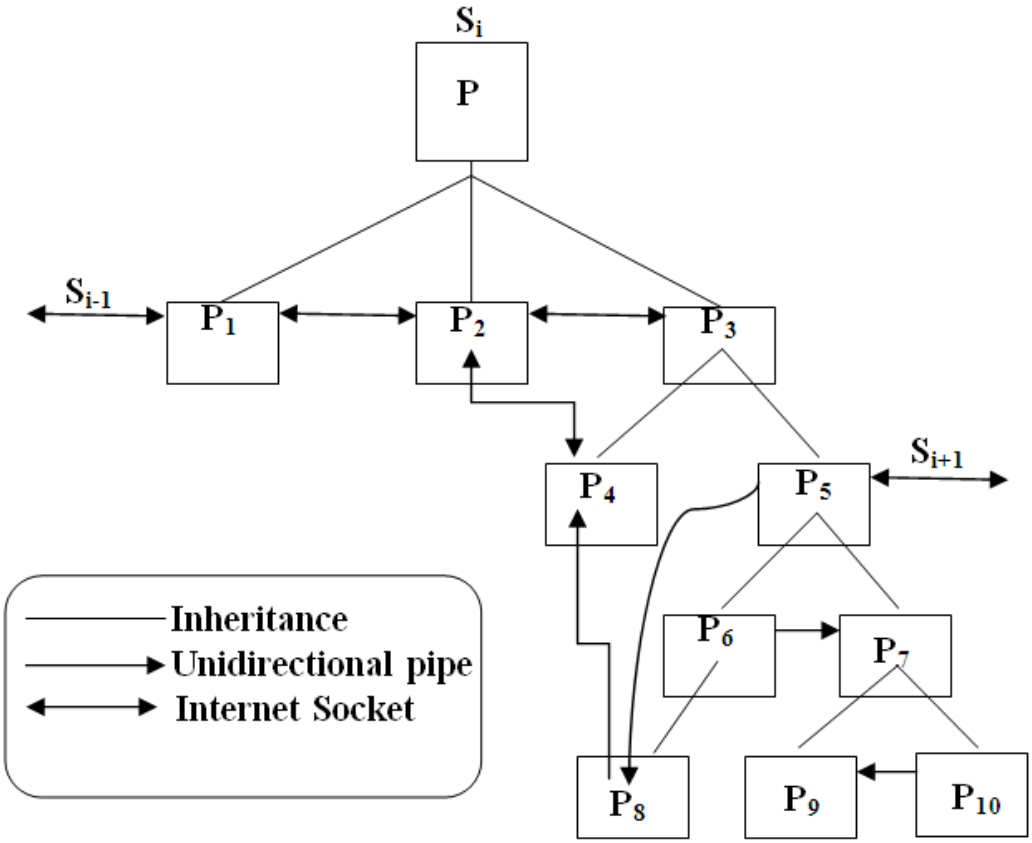

Figure 4. Process structure with 10 different processes

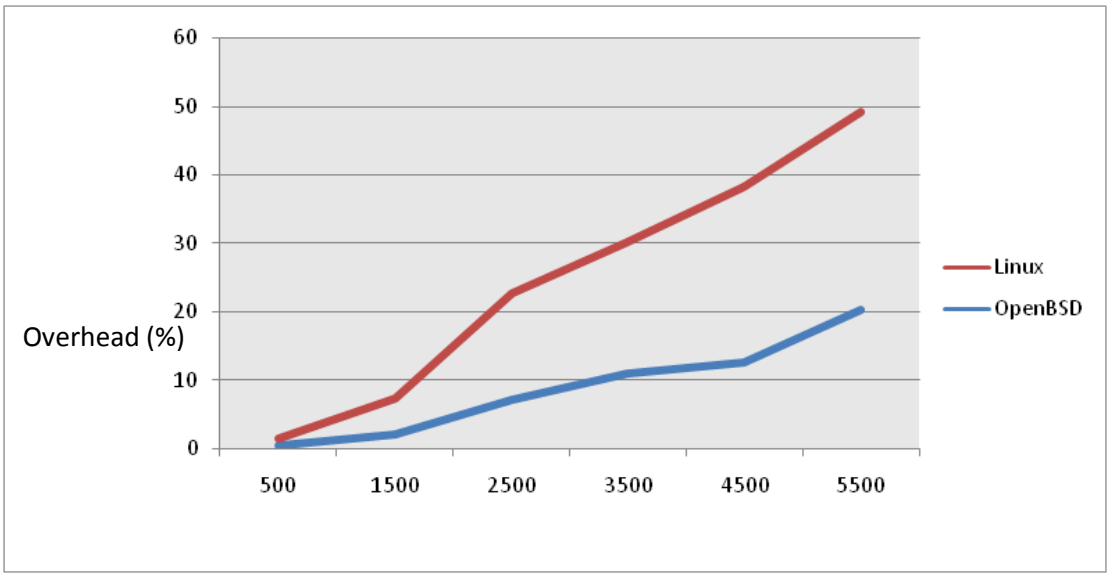

Figure 4. NTP per minute process flow

Comparison with past techniques with NTP performance

Table 3. Comparison with past techniques

Available Techniques/ Flooding types	Hyunjoo Kim et al	Xu Rui et al	Haidar Safaa et Al	Jun-Sang Park et al	Anchit, Wazid, E. S. Pilli	Ruqayya
UDP	No	Yes	No	No	Yes	Yes
HTTP	Yes	No	No	No	Yes	Yes
SYN	No	No	Yes	No	Yes	Yes
SNMP	No	No	No	Yes	Yes	Yes
Random-UDP	No	No	No	No	Yes	Yes
NTP + Random-UDP	No	No	No	No	No	Yes

Table 3, shows comparison of NTP with existing technique, We addressed all the malicious packets with the help of NTP and it can also be ropes for other protocols like TCP, SNMP, SYN, HTTP, etc. Here we are successful to address all BOT packets, for mitigation from these attacks upcoming we will works on authentication algorithms suck as DES, AES, DDA, etc.

5 Conclusion:

In UDP-flooding attack, attacker sends several UDP datagram of unlike sizes at same time. It is similar to chain of connections for systems to hide his or her identity. For forensic exploration in this paper we introduce a new protocol ***Net Token Protocol (NTP)***, which is helpful in network based activity. In this protocol token processing is beneficial as a system chain connection and the protocol has been considered to protect user's privacy by habitual a token which is useful for hash of correlation information. NTP is useful for tracing the UDP chain from the Internet but it not helpful to solving issues, NTP only addresses unwanted or malicious packets with an existing operating system. It can also be supported for other floods like SNMP, HTTP etc.

For future work we are focusing on addressed unwanted packets by NTP, for mitigation of these types of flood attacks, will works and propose some authentication algorithms like DES, AES, DDA, etc.

References

[1] Yuan Tao, Shui Yu," DDoS Attack Detection at Local Area Networks Using Information Theoretical Metrics", 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), 2013.

[2] D. Moustis, P. Kotzanikolaou, "Evaluating security controls against HTTP-based DDoS attacks", 4th IEEE International Conference on Information, Intelligence, Systems and Applications (IISA), 2013.

[3] S. M. Hussain, G. R. Beigh, "Impact of DDoS attack (UDP Flooding) on queuing models", 4th IEEE International Conference on Computer and Communication Technology (ICCCT), 2013.

[4] Alexandru G. Bardas, Loai Zomlot, Sathya Chandran Sundaramurthy,et al , "Classification of UDP traffic for DDoS detection", 5th ACM USENIX conference on Large - Scale Exploits and Emergent Threats (LEET), 2012.

[5] SeRgio S. C. Silva, Rodrigo M. P. Silva, et al, "Botnets: A survey", Elsevier Journal of Computer Networks, vol. 57(2), Feb.2013, pp. 378-403.

[6] Steve Mansfield- Devine, "Battle of the botnets", Elsevier Journal of Network Security, vol. 2010 (5), May 2010, pp. 4-6.

[7] Xu Rui, Ma Wen-Li, Zheng Wen-Ling, "Defending against UDP Flooding by Negative Selection Algorithm Based on Eigenvalue Sets", 5th IEEE/ACM International Conference on Information Assurance and Security, 2009.

[8] K. Argyraki, D. R. Cheriton, "Scalable Network-Layer Defense against Internet Bandwidth-Flooding Attacks", IEEE/ACM Transactions on Networking, vol. 17 (4), 2009.

[9] Jun-Sang Park, Myung -Sup Kim, "Design and Implementation of an SNMP-Based Traffic Flooding Attack Detection System", 11th Springer Asia-Pacific Network Operations and Management Symposium (APNOMS), 2008.

[10] Keisuke Takemori , Masahiko Fujinaga, Toshiya Sayama, et al, "IP Traceback Using DNS Logs against Bots", IEEE International Symposium on Computer Science and its Applications (CSA), 2008.

[11] Haidar Safaa, Mohamad Choumana, Hassan Artailb, Marcel Karama, "A collaborative defense mechanism against SYN flooding attacks in IP network", Elsevier Journal of Network and Computer Applications, vol. 31 (4), Nov. 2008, pp. 509-534.

[12] T. Hurth, F. Mahmoudi "The minimal flavour violation benchmark in view of the latest LHCb data", Elsevier, Volume 865, Issue 3, 21 December 2012, pp. 461-485.

[13] Anil Kurmus, Alessandro Sorniotti, Rudiger Kapitza," Attack surface reduction for commodity OS kernels: trimmed garden plants may attract less bugs", ACM New York, NY, USA ©2011, ISBN: 978-1-4503-0613-3.

[14] Arvind Negi, Punit Sharma, Prasant Chaudhary and Himanshu Gupta. "New Method for Obtaining Digital Signature Certificate using Proposed RSA Algorithm", International Journal of Computer Applications 121(23):24-29, July 2015.

[15] Vural, H. S. Venter, "Using Network Forensics and Artificial Intelligence Techniques to Detect Bot-nets on an Organizational Network", 7th IEEE International Conference on Information Technology: New Generations (ITNG), 2010.

[16] Anchit Bijalwan, Mohammad Wazid, Emmanuel S. Pilli, R.C. Joshi. "Forensics of Random-UDP Flooding Attacks", JOURNAL OF NETWORKS, VOL. 10, NO. 5, MAY 2015.