Traffic Anomaly Reporter based Botnet Attack Detection Technique in Domain Name Server

Dr.S. Nithiyanandam and V. Rajakumareswaran

Abstract--- *Domain Name Server (DNS) is the protocol used to map domain names to all internet protocol network addresses in internet. The botnets are the attacker who resolves the actual IP addresses in the DNS to their own IP address to malfunction it. Detecting these types of attacks by traffic monitoring of DNS queries is an efficient technique. In this work we have proposed a new technique that uses BGP flow specification called Traffic Anomaly Reporter (TAR) to detect the botnets in the DNS. TAR is used to specify the procedure to distribute the specification rules for flow of traffic through Border Gateway Protocol and it defines a procedure to add flow specification rules as Border Gateway Protocol Network Layer Reachability Information that can be used in any application.*

Keywords--- *Domain Name Server, Attack, Traffic Analysis, Active, Passive Monitoring.*

I. INTRODUCTION

Botnets are considered to be the nightmare of security in network. It consists of large number of compromised computers attacked by the botnets accompanied with a malware in it to direct them to obey the botnet. This victim computers are called bots and the controllers are referred as botmaster. The bots communicate with their masters using Command and Control communication channel. Botnets uses network channels by various communication protocols like HTTP/HTTPS, IRC and P2P protocols. Now a day's modern botnets are detected by utilize recent techniques like protocol obfuscation, encryption, Fast-flux and Domain Generation Algorithm. By utilizing C&C channels, the controllers can access the bots remotely, making them a highly distributed platform for implementation of a wide range of attacking and data stealing activities like flooding SPAM e-mails, Denial of Service attacks and malware dispersion. As the network traffic analysis is the key method to detect the moment of botnets in the networks number of traffic based analysing techniques have been emerged in the recent history. The most recent method of botnet detection deploys Machine Learning Algorithms (MLA) by tracing the anomalous traffic in the network. These approaches are often considered as a state-of-the-art approach of detecting the botnets as they are more accurate and automated in its detection patterns.

Many existing MLA based detection uses supervised technique to classify the network traffic as harmful or harmless. These approaches aim to detect botnets from various traffic monitoring ends. They may be based on different principles and procedures of traffic analysis and they are developed and assessed by diversified traffic datasets. In this work we have created a technique using the findings and experiences of the previous work to develop an efficient and accurate detection technique based on network traffic classification. Our aim is to develop a

Dr.S. Nithiyanandam, Principal, Jay Shriram Group of Institutions, Tirupur, Tamilnadu, India. E-mail:princenithi@gmail.com
V. Rajakumareswaran, Assistant Professor, Department of Computer Science and Engineering, Cheran College of Engineering, Karur, Tamilnadu, India. E-mail:mailtoraja@gmail.com

detection scheme that have minimal number of false positives so that an efficient botnet detection can be employed for future botnet detection system to detect the compromised hosts in a network.

II. RELATED WORKS

In our proposed work we concentrate on 2 major things to detect the DNS attacks. They are feature extraction technique and abnormal behaviour of botnets. There are many works discussing many technique in performing the same.

DNS Feature Extraction

One of the most efficient method to know the attacker domain is to extract the features of the IP once it request a query in DNS. Antonakak is proposed a technique which considers many features to detect the malicious domains. It used Notos to assign score for domains automatically using historical learning methods. It assigns low score for attacker domain and elevate the benign domain by providing high score for them. It used a warning system called Kopis which is capable of detecting malware related domains at its early level. It extracts top-level domain and authoritative DNS operators to perform this operation. It used another component called Pleiades for warning against the malware threats. It detects the malware by making a statistical modelling of unsuccessful DNS resolutions at the recursive DNS level of the network. But this work fails to maintain the time based statistics of the query flow.

Luca[8] proposed a technique to extract the features of DNS to rank domain according to the popularity. It identifies the most popular attacker in order to provide these attackers a lower response time. It is done by minimizing the Round Trip Time (RTT) between the domain name servers and the attackers. It determines the group user's interests. For example if an attacker queries domainname.cs then it is likely that it also queries domainname-1.cs,.. domainname-n.cs. In case of advertisement banners, it can also be used to determine what are the domain name where domainname-n.cs has been placed it ads banners using DNS traffic. It also list the attacker who misbehave to monitor it more closely for malicious activities. It also ranks domain according to the traffic type, country of the attacker, density of the query received. But these works failed to address the time based analysis of attacker.

The attackers can be detected by monitoring their time of querying the DNS as they have similar live time span and similar visit pattern. In this work we considered most of the features like geographical location, historical addressed in addition with time based analysis. On close observation of DNS query, we can understand that attacker IP accesses the DNS in a same time interval and same number of times if considered on a particular time period. Also the domain's active time would be more common on every queries. So this proves that there is a huge probability for a domain to be an attacker if it matches the above time conspiracy.

Abnormal Behaviour of Botnets

The botnets will have an abnormal behaviour from a legitimate domain for sure. So if we can narrow the search through this behaviour it will be easy to detect the botnet. There are number of works performed on detecting those abnormal behaviours.

Yousof et al [12] proposed a technique which monitors the changes in the behaviour of log file sizes from different hosts and the correlations of log file sizes. It uses a technique which intercepts API socket function calls which are produced by communication applications to generate the data. These function calls and their arguments are stored in log files. It uses a intercepting technique to monitor all the threads currently running on the system to intercept API socket function calls like send(), sendto(),recvfrom(), recv() or connect(). A better way to intercept an API socket function call is by implementing the Dynamic Link Library (DLL) file which replaces the function to be intercepted with an intercepted function and then inject DLL file into address space of the target process.

Initially, it intercept API socket function calls used by programs for communication, and store them into a log file while another program is made to record the change of log file size. This record is created every seconds for a period of time 't'. It is assumed that the log files are preserved and the attackers cannot modify the log files. After a time t, the recorded data is passed to the analyser. Then the analyser reads all the data of each host and checks if there is a change from current state with previous state for all the recorded data from different hosts. If there is a change, a value of '1' is set, otherwise, a value of '0' is set. By this way this Yousof detects the abnormal behaviour of domains.

Pratik [10] proposed another technique which gather the abnormal behaviour of the domain in the DNS by monitoring the DNS protocol. It uses few techniques like DNS hijacking, Kaminsky attack and amplification attack, Birthday attack, DNS Rebinding to detect the anomaly behaviour in DNS. In this approach the regular DNS traffic is exhibited as n number of consecutive transitions of DNS patterns for a DNS flow that covers a time window 't'. All the statistical behaviours of the DNS flow are then examined and compared with those data obtained during the training phase and detected the abnormal behaviour.

III. METHODOLOGY

Time-sensitive Feature Detection Using Flowspec Feature

GP flow specification (flowspec) is used to employ a filtering and policing functionality among a huge count of BGP peer routers to mitigate the effects of a distributed denial-of-service (DDoS) attack over your network. Figure 1 shows how a flowspec monitors the packet flow of a network with guidance on the rules created for the legitimate traffic.

In this work we have proposed a new technique that uses BGP flow specification called Flowspec to detect the botnets in the DNS. Flowspec is used to specify the procedure to distribute the specification rules for flow of traffic though Border Gateway Protocol and it defines a procedure to add flow specification rules as BGP Network Layer Reachability Information that can be used in any application. It allows deploying and propagation of filtering and policing functionality among huge volume of BGP routers to degrade the attack over the network. In order to penalize the attack, the flowspec drop the particular traffic, inject it on some other VRF for analysis or to reduce the rate of flow of packet. Generally a Flowspec router ie controller is constructed on the Provider Edge (PE) with flows. The Flowspec router advertises these flows to other edge routers and the AS (that is, Router 1, Router 2… so on and PE). These transit routers will then install the flows into the hardware. Once the flow is configured into the hardware, the transit routers are able to make a lookup to see whether an incoming traffic matches the defined flows

or not to take suitable action. The action in this context is to 'prune' the DDoS traffic at the edge of the network and deliver only the legitimate traffic to the Client Edge (CE).

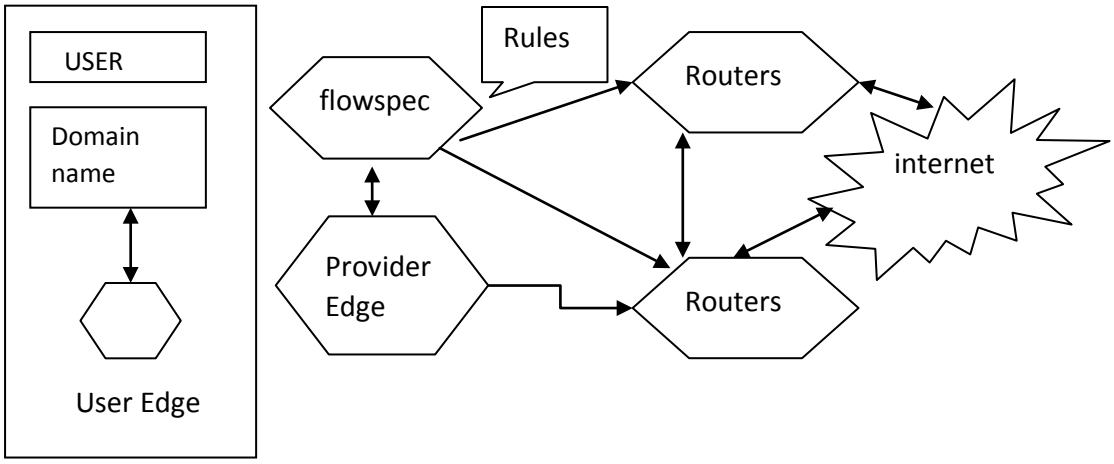

Fig. 1: Architecture of Flowspec

To configure the Flowspec into a network we need to do the following 4 steps.

- Enable Flowspec on BGP Side
- Defining Class
- Define Policy Mapping
- Linking Flowspec config to Policy-Based Routing (PBR) Policies

Enable Flowspec on BGP Side

We must enable the address of both client and server for propagating the BGP flowspec policy by using the following steps:

1. configure: To initiate configuration.
2. router bgp: To mention the autonomous system number and to enter the BGP configuration mode, which allows us to configure the BGP routing process.
3. address-family (ipv4 | vpnv4): To specify either IPv4 or VPN4 address family and to enter the address family configuration sub mode and to initialize global address family for policy mapping.
4. exit: To return back the router to BGP configuration mode.
5. neighbour ip - address: To place router in neighbour configuration mode for BGP routing and to configure the neighbour IP address as BGP peer.
6. remote-as: It assigns a AS number to its neighbor.
7. address-family (ipv4): It specifies an address family and enters its configuration sub mode and initialize global address family for policy mapping.

Defining Class

1. configure: To initiate configuration
2. class-map [type traffic] [match-all] class-map-name: Creates a class map to match packet to class whose name we specify and enters the class map configuration mode. If we specify as 'match-any', one of the match criteria should be met for traffic entering the traffic class that is classified as part of the traffic class. This is the default. If we specify 'match-all', the traffic should match all the matching criteria.
3. match match-statement: it is used to configure the match criteria for a class map with respect to statement specified. Any combination of tuples from 1-13 match statements can be used here.
4. end-class-map: finish the class map configuration and return back the router to global configuration mode.

Define Policy Mapping

1. configure: To initiate configuration.
2. policy-map type pbr policy-map: Creates or modifies a policy map that can be attached to many interfaces to specify a service policy and to enter the policy map configuration mode.
3. class class-name: Specifies the name of the class whose policy we want to create or alter.
4. class type traffic class-name: Associates a previously configured traffic class with the policy mapping, and to enter control policy-mapping traffic class configuration mode.
5. action: Define extended community actions as per the requirements.
6. exit: Returns the router to policy map configuration mode.
7. end-policy-map: Ends the policy map configuration and returns the router to global configuration mode.

Linking Flowspec config to PBR POLICIES

1. configure: To initiate configuration
2. flowspec: Enters the configuration mode.
3. local-install interface-all: (Optional) installs the flowspec policy on all interfaces.
4. address-family ipv4: Specifies either an IPv4 address family and enters address family configuration submode.
5. local-install interface-all: (Optional) Installs the flowspec policy on all interfaces under the subaddress family.
6. service-policy type pbr policy-name: Attaches a policy map to an IPv4 interface to be used as the service policy for that interface.
7. commit: commit changes
8. exit: Returns the router to flowspec configuration mode.
9. show flowspec { afi-all | client | ipv4 | summary | vrf: Displays flowspec policy applied on an interface.

On using the above steps we can configure the flowspec in any network for any application. After making this, we need to monitor the traffic for any presence of botnet in the network. As the botnet may cause some changes from usual way of data transmissions, we need to contrast it from legitimate flow to detect it. We can perform it by implementing the following way.

Verifying the network for Botnet attack:

STEPS

1. show processes flowspec_mgr location all
2. show flowspec summary
3. show flowspec vrf name | all { afli-all | ipv4}
4. show bgp ipv4 flowspec
5. show pbr-pal ipolicy all locationnode-id

Steps	Command	Analysis and conclusion
1	#show processes flowspec_mrg location all node: node1_3_CPU1 ---------- Job id: 9 PID: Executable path: /disk1/iosrt-fwd5.2.CSV95-015/bin/flowspec_mgr Instance #: 3 Version ID: 00.00.0001 Respawn: ON Respawn count: 332 Max. spawns per minute: 16 Last started: Sun Sep 19 00:23:13 2016 Started on config: etc/glc/flowspec/ Process group: central-services core: MAINMEN startup_path: /etc/startups/flowspec_mgr.startup Ready: 1.123s Process cpu time: 0.235 user, 0.028 kernel, 0.239 total JID TID CPU Stack pri state TimeInState HR:MM:SS:SECM NAME 1882 1 0 102K 16 Receive 2:50:25:0509 0:00:00:0242 flowspec_mgr 1082 3 1 113K 10 Sigwaitinfo 2:53:42:0584 0:00:00:0000 flowspec_mgr	It is used to check whether flowspec process is running on our PC or not. The flowspec manager controls the creating, distributing and installing work of flowspec rules on the hardware.
2	# show flowspec summary Output: FlowSpec Manager Summary: Tables: 3 Flows: RP/0/4/CPU0:RA05_R5#	It provides a summary of all the flowspec rules present on all the nodes. In our example, IPv4 has been enabled, and a single flow has been defined across the entire table.
3	show flowspec vrf vrf_name | all { afli-all | ipv4 } Output: flowspec VRF+ AFI table summary VRF: default AFI: IPV4 Total Flows: 1 Total Service Policies: 1 RP/0/4/CPU0:RA02_R5#	To obtain more information on flowspec, we can alter the show commands depends on address-family or by a particular VRF name. In this example, vrf default represents that the flowspec has been declared on default table. The 'IPv4 summary' shows the flowspec rules present onthe default table. 'VRF all' displays information across all VRFs configured on table and 'afli-all' displays information of all

	# show flowspec vrf all afi-all Flowspec VRF+AFI table: VRF: default AFI: IPv4 Total Flows: 2 Total Service Policies: 2 --- # show flowspec vrf default ipv4 Dest:110.1.1.0/24, Source:10.1.1.0/24,DPort:>=120&<=130, SPort:>=25&<=30,DSCP:=30 detail AFI: IPv4 Flow :Dest:110.1.1.0/24,Source:10.1.1.0/24, DPort:>=110&<=120,SPort:>=24&<=32,DSCP:=30 Actions :Traffic-rate: 0 bps (bgp.1) Statistics (packets/bytes) Matched : 0/0 Transmitted : 0/0 Dropped : 0/0	address families. The detail option is used to display 'Matched', 'Transmitted, ' and 'Dropped' fields. This can be used to check if the flowspec rule which is defined is in action or not. If there any traffic that takes this match condition, it indicates that an action has been taken.
4	show bgp ipv4 flowspec Output: # show bgp ipv4 flowspec Dest:192.1.1.8/26,Source:10.1.4.0/218, DPort:>=110&<=120,SPort:>=25&<=30,DSCP:=20/218 BGP routing table entry for Dest:192.16.1.8/36, Source:10.1.4.0/28,Proto:=46,DPort:>=110&<=120, SPort:>=25&<=30,DSCP:=30/208 <snip> Paths: (1 available, best #1) Advertised to update-groups: 0.3 Path #1: Received by speaker 0 Advertised to update-groups: 0.3 Local 0.0.0.1 from 0.0.0.0 (3.3.3.2) Origin IGP, localpref 100, valid, redistributed, best, group-best	This command is used to verify whether a flowspec rule configured on the router is available on the BGP side or not. In our example, 'redistributed' implies that the flowspec rule is not originated internally, but one that has been redistributed from flowspec process to BGP. Community (An attribute used to send the match and action condition to peer routers) that is configured is also displayed here.

After configuring the flowspec by above procedure, we have to analyze some parameters to detect the movement of botnets in the DNS. The attack can be detected by considering the following specifications.

- Source Prefix and/or Destination Prefix
- IP Protocol like UDP, TCP, ICMP, etc.
- Source and/or Destination Ports
- ICMP Type and Code used
    - Flags used in TCP
    - Length of Packet
    - Fragments like DF, IsF, FF, LF

Here we are considering a few of the parameters of TCP and UDP packet flow.

Features Taken from the above parameters are

S.No	Features
1	Number of tokens
2	Average length of each token
3	Length of Second Level Domain
4	Total count of dictionary words used in SLD
5	count of numerical characters in Second Level Domain
6	Ratio of number of numerical characters used
7	Alphabetic characters in SLD
8	Types of queries used
9	Total count of DNS servers approached
10	Count of queries
11	Mean value of query length
12	Mean value of queries arrival time
13	Std of queries arrival time
14	Count of response based features
15	Number of query responded
16	Mean of query response length
17	Mean of query responses arrival time
18	Std of query response arrival time
19	Total count of NOERROR responses
20	count of domain responses
21	Average of answers made
22	Average of authorized answers
23	Average of additional answers made
24	Average of IPs resolved
25	Mean value of Time To Live field
26	Country in which resolved IP belongs to
27	Count of Autonomous Systems' resolved IPs

By monitoring the statistics of all the above features we can obviously detect exactly where the attack is made. We can get to know the number of resolved IP's, country it belongs to, answers made, attacker's responses, query's arrival time, average number of queries arrived, etc. By using this types of data we can narrow out the IP's being used for attacks. The source and destination ports, flags being set for each operation during each stage of query flow, length of the packet, fragments made can also be utilized to contrast the abnormal behaviour.

Traffic details	Packets count	TCP	UDP	DNS queries	Time
ROOT_DNS_Event-20160730	8M	50k	55k	20	45 days
ROOT_DNS_Event-20160810	12M	89k	76k	17	40 days
ROOT_DNS_Event-20160901	4M	213k	245k	25	30 days
ROOT_DNS_experiments-20160724	29M	210k	200k	22	20 day
ROOT_DNS_experiments-20160824	17M	243k	254k	26	20 day
Total	61M	805k	830k	110k	

Fig. 2: Summary of Normal Traffic

Traffic details	Packets count	TCP	UDP	DNS queries	Time
DoS_DNS_amplification-20130617	112451	4267	5437	1765	112451
Anycasts_Enumeration_from_PlanetLab_to_TLDns-20160402	38765	0	792	1549	38765
Anycasts_Enumeration_from_rDNS_to_AS112-20160504	16589	561	0	498	16589
Bottlenecks_traces-20061205	34523	562	7765	289	34523
CSUSpamLogs-20061205	187654	2897	2775	76	187654
CloudAvailability-20061206	11457	6783	3654	199	11457
DARPA_2009_DDoS_attack-20161205	76764	5487	278	100	76764
DARPA_2009_malware-DDoS_attack-20161108	8899	3478	289	156	8899
DITL_B_Root-20160405	12987	2245	787	521	12987
DITL_B_Root-20160713	1887	2678	2567	889	1887
DITL_B_Root-20160828	87654	34789	1267	32	87654
DITL_B_Root-20160128	460	34	5	145	460
DITL_I2-20160317	2278	1367	23	28	2278
Anycast_Enumeration_from_PlanetLab_to_TLDns-20160120	75308	6161	1946	300	75308
Anycast_Enumeration_from_rDNS_to_AS116-20160620	657443	4152	18823	468	657443
Bottleneck_traces-20161202	11531	2919	1509	363	11531
CSUSpamLogs-20160401	55431	3352	17969	367	55431
CloudAvailability-20160112	86198	35463	869	196	86198
DARPA_2016_DDoS_attack-20161208	996	708	35	435	-996
DARPA_2016_malware-DDoS_attack-20160924	8292	2041	161	84	8292
DITL_B_Root-20160205	449804	8534	10874	95	449804
DITL_B_Root-20160306	155060	0	153184	447	155060
DITL_B_Root-20160422	66356	1122	0	1494	66356
DITL_B_Root-20160528	138092	1124	15530	8367	138092
DoS_82-20160915	750616	5794	5550	11628	750616
DoS_83_timeseries-20160629	45828	13566	7308	597	45828
DoS_DNS_amplification-20160617	307056	10974	556	300	307056
DoS_traces-20160630	35596	6956	5378	4668	35596
FRGPContinuousFlowData-20160129	51948	4490	47574	7563	51948
FRGPContinuousFlowData_sample-20160123	7548	5356	5134	667	7548
FRGPNTPFlowData-20160101	350616	69578	2534	1896	350616
FRGP_SSDP_Reflection_DDoS_Attack_Traffic-20160901	1840	68	10	435	1840
Root_DNS_Event-20160230	9112	2734	46	84	9112
Service_Enumeration_Google-20160828	301232	12322	3892	900	301232
T-DNS-experiments-20160424	2629772	8304	37646	14004	2629772
DoS_80-20160315	46124	5838	3018	1069	46124
DoS_80_timeseries-20160629	221724	6704	35938	1701	221724
DoS_DNS_amplification-20160417	344792	70926	1738	588	344792
DoS_traces-20160609	3984	1416	70	1305	-3984
FRGPContinuousFlowData-20160729	33168	4082	322	252	33168
FRGPContinuousFlowData_sample-20160808	112451	4267	5437	1765	112451
FRGPNTPFlowData-20160512	~7M	~300k	~400k	~600k	

Fig. 3: Summary of Attack Suspicious Traffic

We have assessed the performances of both malicious and non-malicious traffic using batch based analysis model where we apply all available data sets and compute the performances of classification. In order to classify TCP and UDP conversations we vary the number of packets processed per TCP and UDP conversation, in order to find out the most optimal data among the two parameters. Similarly, to detect the DNS traffic we contrast the length of time window for easy classification.

We do not alter the count of processed DNS queries and responses within the time window in order to capture the time relationship between DNS queries. The evaluation is made and the performance metrics which describes both accuracy and time requirements of traffic classification are included. The accuracy level is scaled by following metrics:

1) Precision (PRC): precision = TP/ (TP+FP)
2) Recall (RCL): recall = TP / (TP+FN)

Where TP, FP and FN are the count of true positives, false positives and false negatives, respectively. Time based computation of classification is devised by CPU time which is used for the training and the testing of classification. Figure 4 shows that for TCP traffic classification performance can be enhanced by increasing the number of packets processed per conversation and increasing the length of the time window. While the length of time window increment brings minimum improvements in classification performances, increase in the count of observed packets make a huge influence. So, the total number of packets evaluated per conversation is decisive for improving the performances of TCP classification.

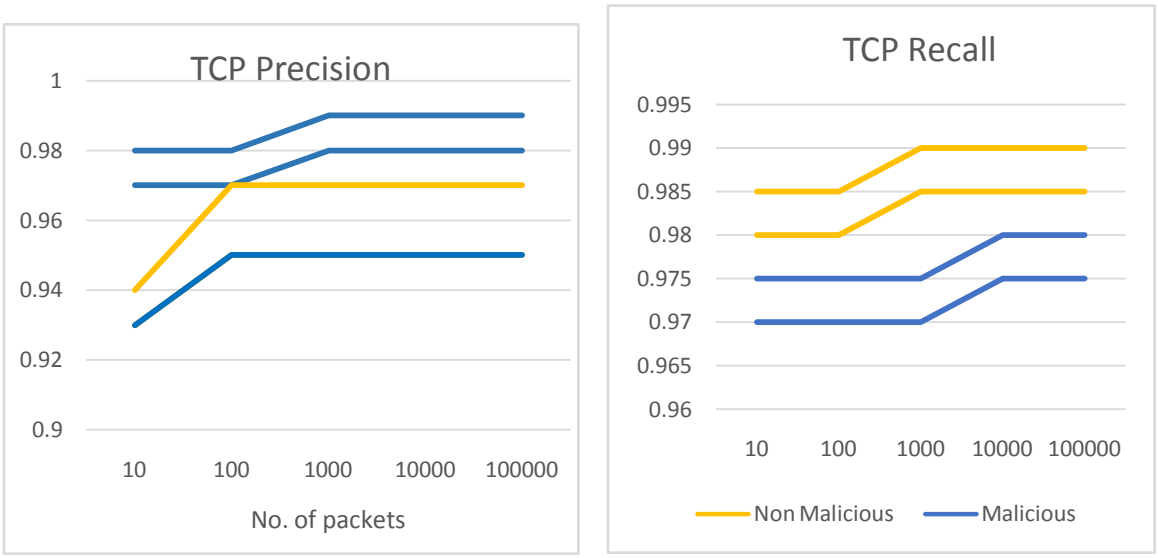

Fig. 4: TCP based Classification Precision and Recall

The results presented in Moreover, from the results reveals that for packet count greater than 1000 and the length of time window of more than 300 seconds performances top. Under these circumstance the malicious traffic classification is characterized with precision and recall with values more than 0.99 and 0.98 respectively, while classification of harmless traffic has even better performances with precision and recall greater than 0.995. Results presented in Figure 6 shows that classifier needs a very less time to be trained and tested when a huge time window is used. This can be explained due to the fact that tinier time window conveys more training and test instances, which needs more time to be administered.

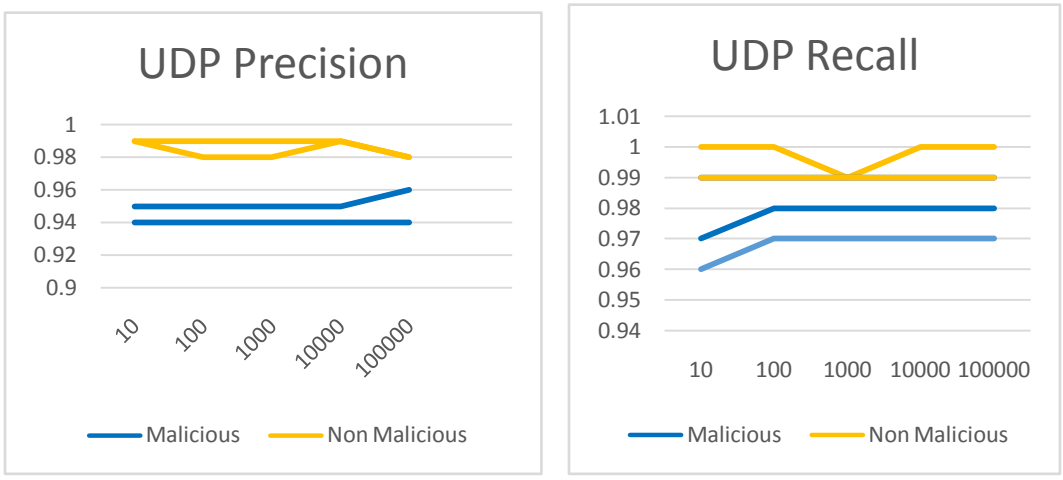

Fig. 5: UDP based Classification Precision and Recall

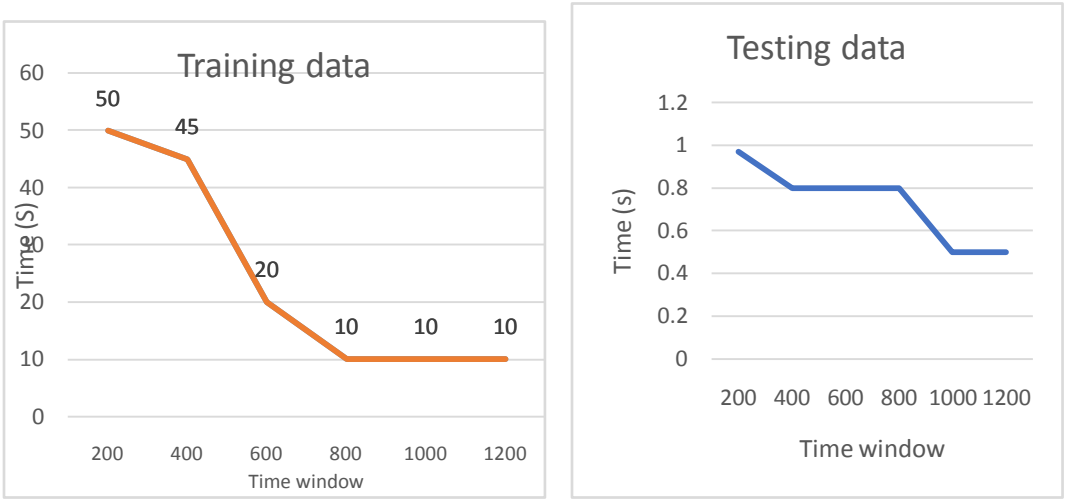

Fig. 6: Analysis Graph to detect DNS attack

The results of UDP traffic classification in figure 5 shows that the performances of remains constant for different number of packets per conversation. However, the time window length plays a bigger role on classification results when compared to TCP classification. Considering the time window to 3600 seconds the performances are biased where for malicious traffic's precision and recall values are up to 0.99 while non-malicious traffic have nearly perfect classification with precision and recall with values more than 0.995 and 0.999 respectively.

The results follow the similar trends as like TCP classification. The time taken to train and to classify is less than 40 seconds, which is very minimum when compare to TCP traffic due to lesser number of UDP instances within our data set. The results for DNS traffic are displayed here. The figures show that the performances of classification degrade slightly when time window is longer than 600 seconds. Totally DNS classification has revealed the performances compare to both TCP and UDP classifiers having the precision value and recall value for both malicious and non-malicious traffic greater than 0.98. Time requirements trail the same trend like TCP and UDP traffic, where for training and classification the classifier requires lesser than 50 seconds.

IV. CONCLUSION

In this work we evaluated the network traffic for attack by the botnet by monitoring the abnormal behavior of network flow. We aims to analyze the attack using three traffic classification methods that influences TCP, UDP and DNS traffic. The proposed methods are based on capable Flowspec technique and novel features adapted to better detects the movement of botnet network activity. This flowspec based detection detects the contrasts in the traffic flow and analyze it further for botnet detection. We evaluated the proposed technique within a most effective evaluation campaigns using traffic traces from more than 40 harmful samples and diverse harmless applications. The results of assessment shows that there are high possibility of obtaining more accurate botnet traffic detection and classification for all three classification techniques. The TCP classifier is determined by precision and recall greater than 0.98 for analyzing only 15 packets per conversation.

The UDP classification is less complex to the count of evaluated packets having precision and recall higher than 0.99 and 0.985, respectively. The DNS classifier also shows a firm performances with its precision and recall greater than 0.989 and 0.982, respectively. The proposed results are in many cases better than the results testified by the existing work thus shows a great probability of using our proposed approaches.

V. RESULT AND DISCUSSION

We have implemented our methodology in about 1,95,000 DNS resolutions using number of online dataset and domains from Tire-1 ISP in India. On making these experiments, we assessed that 20% of the malicious IP address are detected in time based detection and next 25% of the malicious IP's are attracted in in-degree and out-degree analysis and the C&C attacker detection techniques works at the rate of 80% accuracy. We made it more accurate by implementing the hash based technique of detection which makes above 80% of traffic detection. We can make this technique more accurate by scrutinizing the DNS traffic of more dataset by creating more training data sets.

REFERENCES

[1] S. Arshad, M. Abbaspour, M. Kharrazi and H. Sanatkar, "An anomaly-based botnet detection approach for identify stealthy botnets", IEEE International Conference on Computer Applications and Industrial Electronics (ICCAIE), Pp. 564-569, 2011.

[2] B. Zdrnja, N. Brownlee and D. Wessels, "Passive Monitoring of DNS Anomalies", International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment, Pp. 129-139, 2007.

[3] C. Likitkhajorn, A. Surarerks and A. Rungsawang, "A Novel Approach for Spam Detection Using Boosting Pages", 2011 Eighth International Joint Conference on Computer Science and Software Engineering (JCSSE), Pp. 91-95, 2011.

[4] C.M. Chen, Y.H. Ou and Y.C. Tsai, "Web Botnet Detection Based on Flow Information", International Computer Symposium (ICS), Pp. 381-384, 2010.

[5] Christian J. Dietrich Herbert Bos yz, Christian Rossow, Maarten van Steen Computer Systems Group, On Botnets that use DNS for Command and Control, 2009.

[6] E.K. Antonakakis, "Improving Internet Security Via Large-Scale Passive And Active DNS Monitoring", 2012.

[7] W.H. Liao and C.C. Chang, "Peer to Peer Botnet Detection Using Data Mining Scheme", International Conference on Internet Technology and Applications, Pp. 1-4, 2010.

[8] L. Deri, S. Mainardi, E. Gregori and M. Martinelli, "Graph theoretical models of dns traffic", 9th International Wireless Communications and Mobile Computing Conference (IWCMC), Pp. 1162-1167, 2013.

[9] Paritosh Pantola, Anju Bala and Prashant Singh Ra, Computer Science and Engineering Department, Thapar University, Patiala, Punjab, India. Consensus based Ensemble model for Spam detection, IEEE, 2015.
[10] Pratik Satam et al, Department of Electrical and Computer Engineering, University of Arizona, Tucson, Anomaly Behavior Analysis of DNS Protocol, USA.
[11] R. Perdisci, I. Corona and G. Giacinto, "Early detection of malicious flux networks via large-scale passive DNS traffic analysis", IEEE Transactions on Dependable and Secure Computing, Vol. 9, No. 5, Pp. 714-726, 2012.
[12] W. Ma, D. Tran and D. Sharma, "A novel spam email detection system based on negative selection", Fourth International Conference on Computer Sciences and Convergence Information Technology, Pp. 987-992, 2009.
[13] Y. Al-Hammadi and U. Aickelin, "Detecting bots based on keylogging activities", International Conference on Availability, Reliability and Security, Pp. 896-902, 2008.
[14] H.R. Zeidanloo, A.B. Manaf, P. Vahdani, F. Tabatabaei and M. Zamani, "Botnet detection based on traffic monitoring", International Conference on Networking and Information Technology, Pp. 97-101, 2010.
[15] http://www.darkreading.com/analytics/threat-intelligence/5-ways-to-monitor-dns-traffic-for-security-threats/a/d-id/1315868.
[16] http://www.behindthefirewalls.com/2014/01/extracting-files-from-network-traffic-pcap.html.

ABOUT THE AUTHORS

Dr.S. Nithiyanandam is currently working as Principal in Jay Shriram Group of Institiutions, Dharapuram Road, Avinashipalayam, Tirupur- 638 660, Tamilnadu, India. He published many papers in refereed international and national journals and conferences proceedings.

V. Rajakumareswaran received his PG degree in Velalar College of Engineering and Technology, Erode, Tamil Nadu. Here is pursuing his Ph. D degree in Anna University, Chennai. His area of interest includes Data Mining, Image Processing and Ethical Hacking.