DEFENDING AGAINST DOS ATTACKS IN CLOUD COMPUTING

Jeba Sinthi. A, Sahaya Reema M,E

ABSTRACT: Cloud computing is a promising information technique (IT) that can organize a large amount of IT resources in an efficient and flexible manner. Increasingly numerous companies plan to move their local data management systems to the cloud and store and manage their product information on cloud servers. An accompanying challenge is how to protect the security of the commercially confidential data, while maintaining the ability to search the data. In this paper, a privacy-preserving data search scheme is proposed, that can support both the identifier-based and feature-based product searches. Specifically, two novel index trees are constructed and encrypted, that can be searched without knowing the plaintext data. Analysis and simulation results demonstrate the security and efficiency of our scheme.

KEYWORDS: Representational state Transfer(REST), Denial of Service(DOS), Distributed DOS(DDOS), Client Puzzle Protocol(CPP), intrusion Detection System(IDS).

INTRODUCTION: Cloud computing is an internet-based computing in which large groups of remote servers are networked to allow sharing of data-processing tasks, centralized data storage, and online access to computer services or resources. Clouds can be classified as public, private or hybrid.Cloud computing or in simpler shorthand just "the cloud", also focuses on maximizing the effectiveness of the shared resources. Cloud resources are usually not only shared by multiple users but are also dynamically reallocated per demand. This can work for allocating resources to users. For example, a cloud computer facility that serves European users during European business hours with a specific application (e.g., email) may reallocate the same resources to serve North American users during North America's business hours with a different application (e.g., a web server). This approach should maximize the use of computing power thus

reducing environmental damage as well since less power, air conditioning, rack space, etc. are required for a variety of functions.The expression cloud is commonly used in science to describe a large agglomeration of objects that visually appear from a distance as a cloud and describes any set of things whose details are not inspected further in a given context. In analogy to above usage the word cloud was used as a metaphor for the Internet and a standardized cloud-like shape was used to denote a network on telephony schematics.

This section defines the characteristics that a DDoS attack against an application server running in the cloud should have to be stealth. Regarding the quality of service provided to the user, we assume that the system performance under a DDoS attack is more degraded, as higher the average time to process the user service requests compared to the normal operation. Therefore, the first requirement to design an efficient DDoS attack pattern is the ability of the attacker to assess the damage that the attack is inflicting to the system, by spending a specific budget to produce the malicious additional load. The attack damage is a function of the 'attack potency', which depends on the number of concurrent attack sources, the request-rate of the attack flows, and the job-content associated to the service requests to be processed. Moreover, in order to make the attack stealthy, the attacker has to be able to estimate the maximum attack potency to be performed, without that the attack pattern exhibits a behavior that may be considered anomalous by the mechanisms used as a protection for the target system. In the following sections, starting from a synthetic representation of the target system, we describe the conditions the attack pattern has to satisfy to minimize its visibility as long as possible, and effectively affect the target system performance in the cloud environment.

LITERATURE SURVEY: In this Paper[1], we present an intrusion tolerant approach for DoS attacks to WSs that exploit XML vulnerabilities. In this work, we consider an attack

as a malicious intentional fault through which an attacker aims to exploit a system vulnerability. Intrusion is a malicious externally induced fault resulting from an attack that has been successful in exploiting a vulnerability. The proposed solution emphasizes the relation among attack detection, intrusion diagnosis and intrusion recovery. In this paper[2], we present our performance analysis of network I/O workloads hosted in different VMs of a single physical machine. First, we study the impact of running idle guest domains on the overall system performance, addressing the cost, and benefit of managing idle VM instances in virtualized data centers. Second, we conduct an indepth measurement analysis of concurrent network I/O applications colocated in a virtualized cloud in terms of throughput performance and resource sharing effectiveness. Finally, we study how different CPU resource sharing strategies among VMs hosted on a single physical machine under different workload rates may impact the overall performance of a virtualized system.

In this Paper[3], Therefore, the biggest challenges of this method are threefold: 1) How to construct a testing matrix to enable prompt and accurate detection. 2) How to regulate the service requests to match the matrix in practical system. 3) How to establish proper thresholds for server source usage indicator, to generate accurate test outcomes. Similar to all the earlier applications of GT, this new application to network security requires modifications of the classical GT model and algorithms, so as to overcome the obstacle of applying the theoretical models to practical scenarios. Provide an end-to-end underlying system for GTbased schemes, without introducing complexity at the network core. In this Paper[4], In this paper, we consider sophisticated attacks that utilize legitimate application layer requests from legitimately connected network machines to overwhelm Web server. Since the attack signature of each application layer DDoS is represented in abnormal user behavior, we propose several mechanisms, which can be used for application DoS/DDoS attack detection.In this Paper[5], we propose generating vulnerability-driven signatures at network level without any host-level analysis of worm

execution or vulnerable programs. As the first step, we design a network-based length-based signature generator (LESG) for the worms exploiting buffer overflow vulnerabilities1. We further prove the attack resilience bounds even under worst-case attacks with deliberate noise injection. As the first step toward this ambitious goal, we propose length-based signature generator (called LESG) which generates length-based signatures which cannot be evaded. That is, even when the attackers know what the signatures are and how the signatures are generated, they cannot find an efficient and effective way to evade the signatures.

In this Paper[6], we propose a novel framework to robustly and efficiently detect DDoS attacks and identify attack packets. The key idea of our framework is to exploit spatial and temporal correlation of DDoS attack traffic. In this framework, we design a perimeter-based anti-DDoS system, in which traffic is analyzed only at the edge routers of an internet service provider (ISP) network. (1) temporal-correlation based feature extraction and (2) spatial-correlation based detection. Our simulation results show that the proposed framework can detect DDoS attacks even if the volume of attack traffic on each link is extremely small.

PROPOSED SYSTEM: This project consists of four modules such as data owner module, data user module, server module and hacker module. The data owner module consists of registration, login, File upload, view the file request from user, verify files, logout. The data user module consists of registration, login, file search, send request to data owner, File download, logout. Server module consists of maintain user details, hacker details, Verify file, service queue model, logout. Hacker module consists of login, File search, insert malicious data, logout. The four modules and their sub modules are listed below:The data owner module consists of following process. The data owner has the first step of registration. (1) The data owner registers their personal details such as their name, mail id, mobile number, password, date of birth and user type. During registration for each data owner having unique ID. The data owner have to fill all details in registration form. Otherwise they will get a error message. And the mail id is in the

form of internet mail id (eg: xxx@gmail.com), name (must be characters), mobile number (it must starts with 7 or 8 or 9 series, It must be 10 digit number). After giving all details only the data owner complete their registration.(2)After registration the data owner will login by using the unique id generated during registration, their name and user type. If the user give anything wrong among the above he/she will get a error message. Otherwise he/she will goto their home page. (3)After Login the user will upload their file from local disk and encrypt their file using the key which is generated randomly using advanced encryption standard. These files are uploaded to server. After verification the file will be uploaded to server.(4)In this module the data owner can view the request from the data user's. The file uploaded by the data owner is needed to data user will send the file request to corresponding data owner with their id and file id. If the data owner can view the request and they have the rights to accept or reject the file request. If the data owner accepts the file means the data user can download that file otherwise they have not have the permission to download that file.(5) This module is used to verify the file uploaded by data owner is hacked by attackers or not. The data owner verify their files periodically, because if the uploaded file by data owner is hacked by any hacker means the server send the alert message to data owner. From the alert message the data owner can recover their files.Logout: After performing all the tasks of data owner module the data owner can logout from their page.

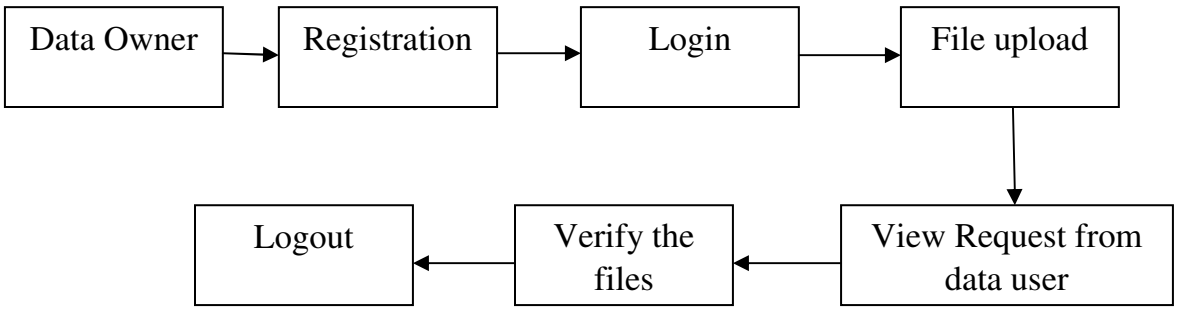

Fig: Data Flow Diagram of Data owner

Data User:The data User module consists of following process. (1)The data user registers their personal details such as their name, mail id, mobile number, password, date of birth and user type. During registration for each data owner having unique ID. And the mail id is in the form of internet mail id (eg: xxx@gmail.com), name (must be characters), mobile number (it must starts with 7 or 8 or 9 series, It must be 10 digit number). After giving all details only the data owner complete their registration.(2) After registration the data user will login by using the unique id generated during registration, their name and user type. (3)After login the data user has the option to search the file by using file name. If the file name searched by the user is already uploaded by the data owner means list of that file details are displayed. For download that full file the data user has to send file request to data owner. The data user sends their request with their id.(4)After getting the file name, file id and data owner id the data user can send the file request to corresponding data owner. (5) After the response from the data owner the data user can download their files. (6)After performing all the tasks of data owner module the data owner can logout from their page.

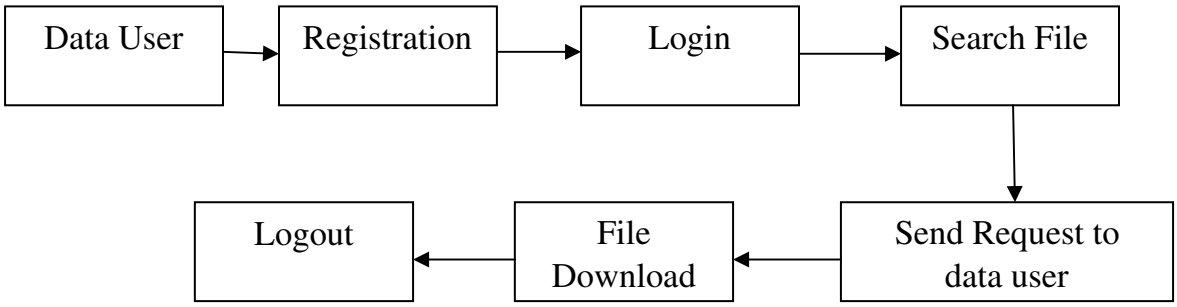

Fig: Data Flow Diagram of Data user

Server:The server module consists of the following: The server will login by using the id and password. The server maintain the details of data owner, hacker details and service queue. The server will maintain the details of data owner and data user details. And also maintain the details of hacker details.The server can verify the files uploaded by the data

owner. After uploading of file by data owner it will goes to server page. The server will verify the files and uploaded to server.

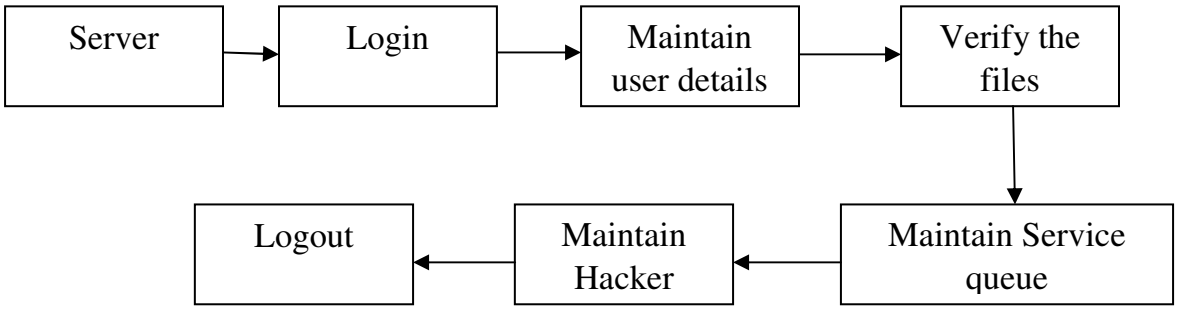

Fig: Data Flow Diagram of Server

Hacker: The hacker module consists of following: Login, Search the file, Insert malicious data.

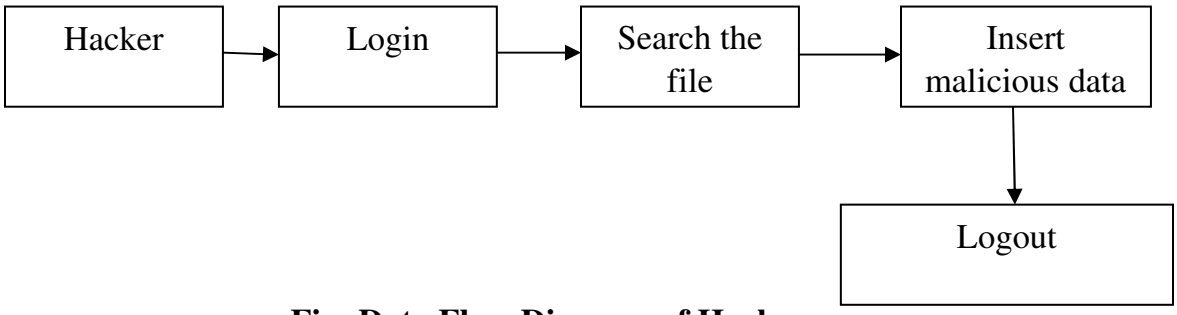

Fig: Data Flow Diagram of Hacker

RESULT: The proposed attack strategy, namely Slowly-Increasing- Polymorphic DDoS Attack Strategy (SIPDAS) can be applied to several kind of attacks, that leverage known application vulnerabilities, in order to degrade the service provided by the target application server running in the cloud. The term polymorphic is inspired to polymorphic attacks which change message sequence at every successive infection in

order to evade signature detection mechanisms. Even if the victim detects the SIPDAS attack, the attack strategy can be re-initiate by using a different application vulnerability (polymorphism in the form), or a different timing (polymorphism over time).

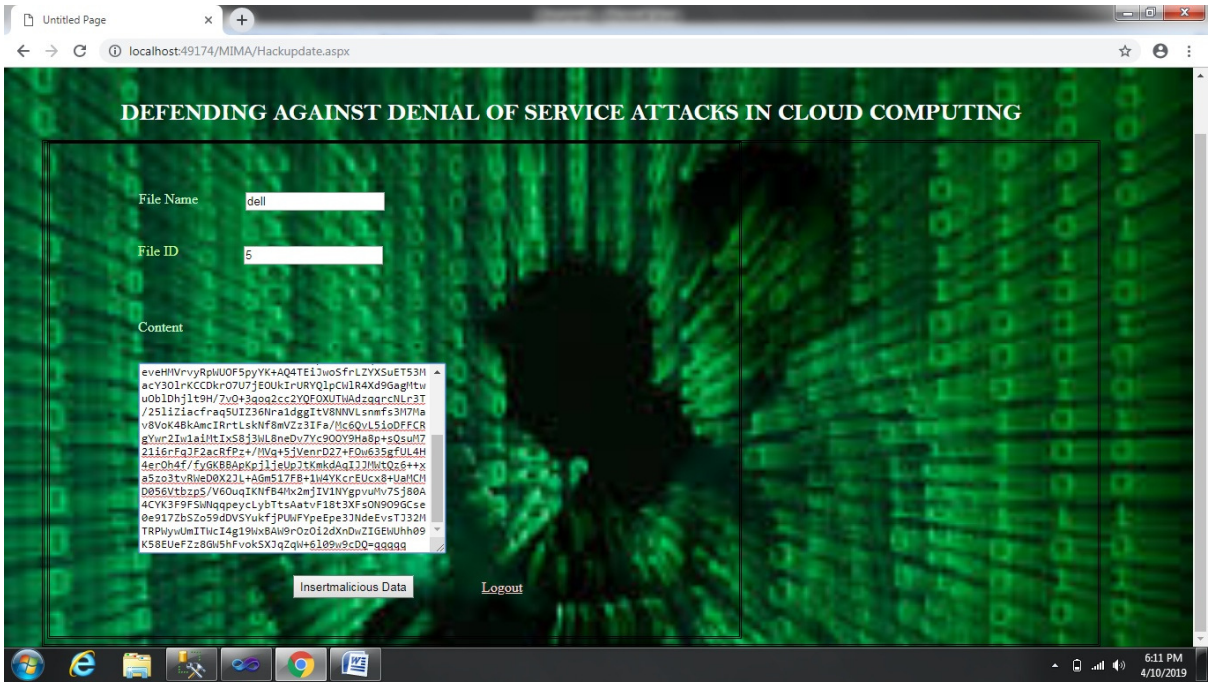

Fig: The hacker will insert malicious data to the displayed result of searched file

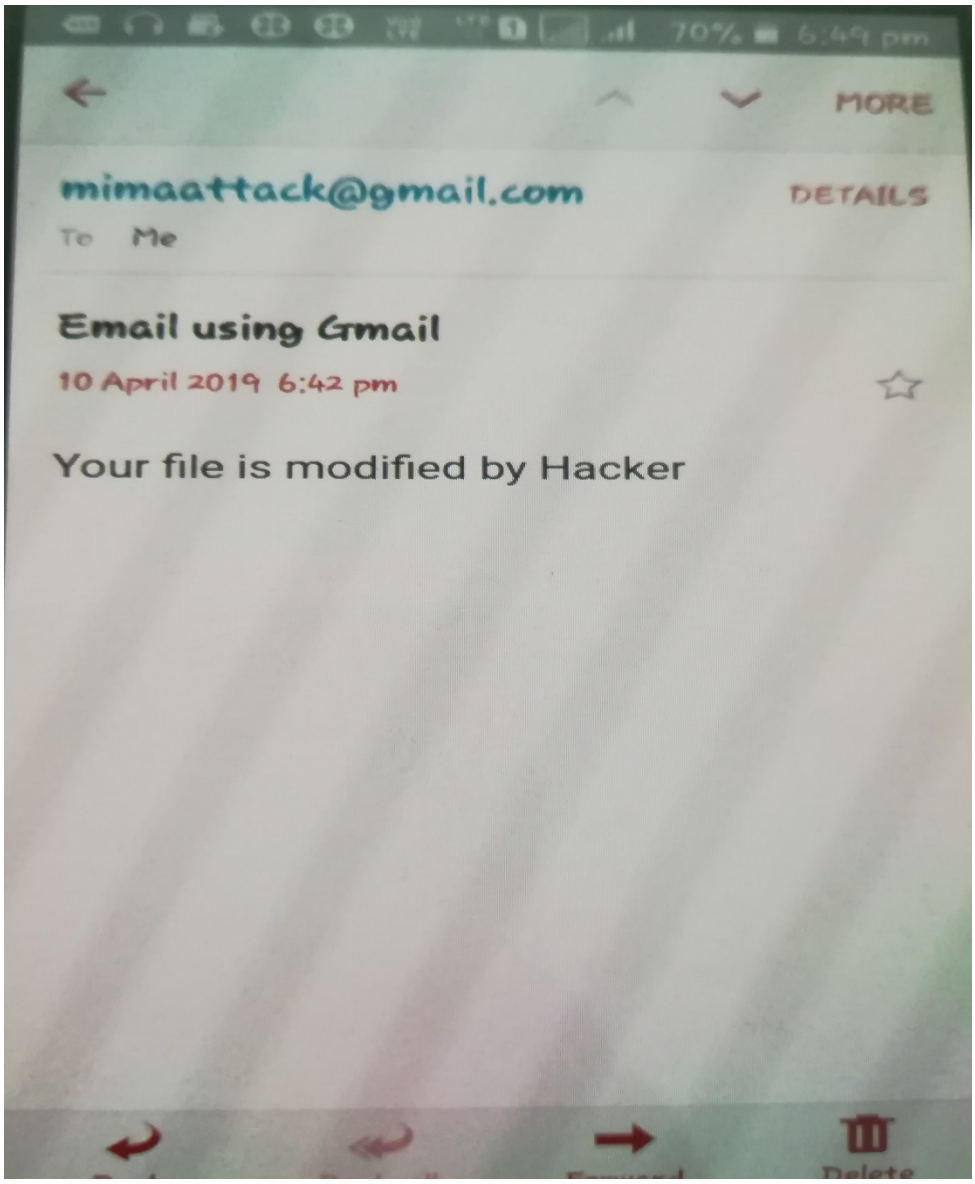

Fig: The server sends the alert messageto data owner

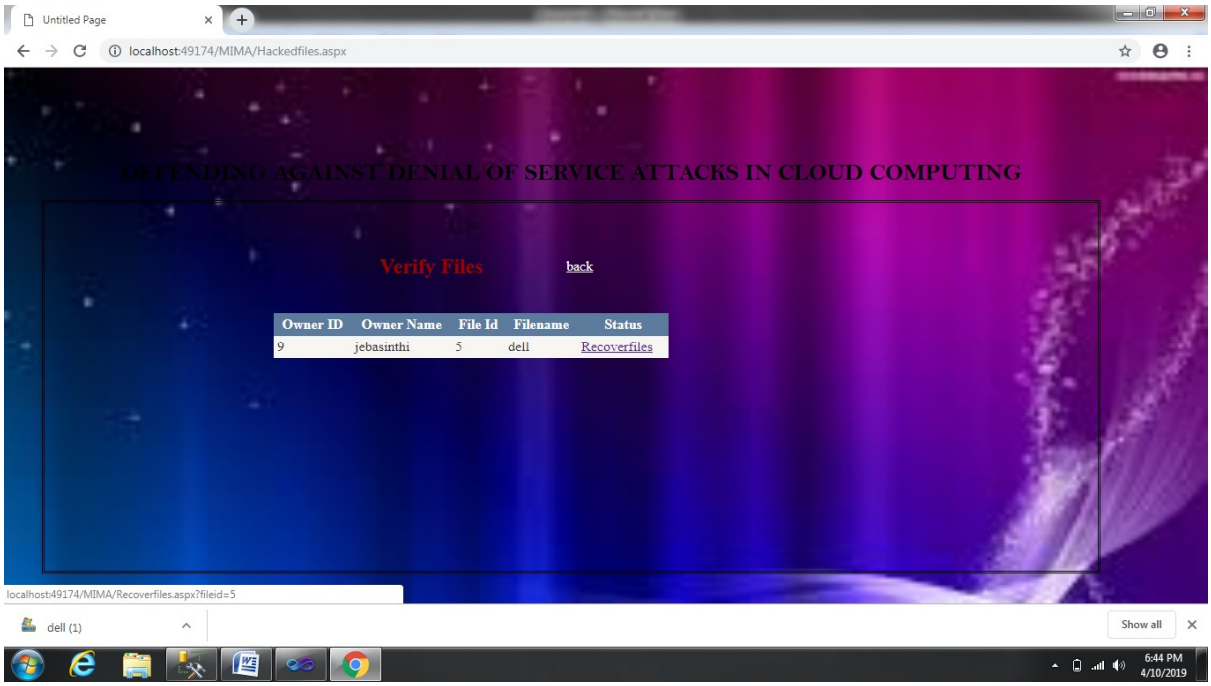

Fig: The data owner view the alertmessage and recover it from the server

CONCLUSION: This project proposes a strategy to implement stealthy attack patterns, which exhibit a slowly-increasing polymorphic behavior that can evade, or however, greatly delay the techniques proposed in the literature to detect low-rate attacks. Exploiting a vulnerability of the target application, a patient and intelligent attacker can orchestrate sophisticated flows of messages, indistinguishable from legitimate service requests. This project consists of four modules they are data owner, data user, hacker and server. The data owner uploads their files on server. The server maintain all the files and details of user's.

REFERENCES: [1] M. Ficco and M. Rak, "Intrusion tolerant approach for denial of service attacks to web services," in Proc. IEEE Int. Conf. Data Compression, Commun. Process., Jun. 2011, pp. 285–292.

[2] Y. Mei, L. Liu, and X. Pu, "Performance analysis of network I/O workloads in virtualized data centers," IEEE Trans. Services Comput., vol. 6, no. 1, pp. 48–63, Jan. 2013

[3] Y. Xuan, I. Shin, M. T. Thai, and T. Znati, "Detecting application denial-of-service attacks: A group-testing-based approach," IEEE Tran. Parallel Distrib. Syst., vol. 21, no. 8, pp. 1203–1216, Aug. 2010.

[4] V. Durcekova, L. Schwartz, and N. Shahmehri, "Sophisticated denial of service attacks aimed at application layer," in Proc. 9th Int. Conf. ELEKTRO, 2012, pp. 55–60.

[5] L. Wang, Z. Li, Y. Chen, Z. Fu, and X. Li, "Thwarting zero-day polymorphic worms with network-level length-based signature generation," IEEE/ACM Trans. Netw., vol. 18, no. 1, pp. 53–66, Feb. 2010

[6] K. Lu, D. Wu, J. Fan, S. Todorovic, and A. Nucci, "Robust and efficient detection of DDoS attacks for large-scale internet," Comput. Netw., vol. 51, no. 18, pp. 5036–5056, 2007.