Improving Security of Data in Cloud using MININET

Ramanathan.A,Karthika.G, Ranjitha.M,
M.EStudent, Department of Computer Science and Engineering,
Chettinad College of Engineering and Technology,
Karur,Tamilnadu.

***Abstract*- Now a day's cloud computing is becoming a favoredtechnology. Many researches are going on Cloud computing environment and usage of this technology is increases tremendously day by day. By the reason of disparate client and network configuration present in cloud network, it is needed to protect each one of them in the cloud network from different attacks. To provide security we can apply existing network security devices but it involves more complexity, dynamism as well as diversity. In this paper we proposed new framework CLOUDWATCHER, which enable us to monitor large and dynamic cloud network. This framework automatically bypasses network packets using open flow technology from preinstalled network devices by this way a cloud network administrator ensures cloud network by inspecting it. We propose two different algorithms which provide dynamic monitoring of network in cloud. To measure the performance of these algorithms we used anomaly intrusion detection system. To evaluate this framework, our proposed framework implemented on MININET environment and evaluated it on different network environment using open flow technology. This framework is quite perceptive, easy to learn and simple to use.**

Keywords-Cloud watcher, Open flow, dynamic computing, CloudSecurity, Data Security.

I. INTRODUCTION

The features of cloud computing explain in following manner, first of all its vast environment which includes many number of host and virtual machines. Amazon EC2 cloud provide service nearly to millions of host, this is possible because every host will be useful to many virtual machine [1]. The structure of cloud environment is little complex. The cloud network can be managed by including many number of machines and number of clients who may need network specifications. Cloud environment is very dynamic that is it changes according to network conditions. Any kind of service needed by consumer is provided by cloud, if necessary it will run bulk of VM's to ensure service. The main advantage of cloud computing is virtual machine in hosts can be added or removed according to their requirement.

Normally, we can ensure security of organization network by using firewalls and NIDS. We can easily use present network security devices to ensure security of cloud environment. There are many issues we have to consider while using security devices in cloud environment.

First, we should take care of all kinds and threats to network. Generally, network is damaged by threats from other networks. To ensure network security from outside threats we can apply security devices between two networks. Another kind of threat to networks is from its own infected internal hosts/VM, this problem largely occurs in public related cloud network [4]. This threat can detect by installing security devices between internal nodes.

Since, we configuration of cloud network is very difficult because of its complex nature. Moving security devices from one location to another, it's not simple task, therefore we should install these security devices carefully. Second problem in cloud computing is virtual machine migration from one host to another host due to its dynamic nature. For example, to monitor network traffic between two hosts we use NIDS on link between hosts. VM in any host can move to another host, so we have to consider the location of NIDS according to movement of VM.

To solve this problem we propose new framework Cloud Watcher, which does following functions:

It manages and ensures security of packets in network by using security devices

It provides help to people for using services by defining simple policy language.

In this framework, we use software defined networking i.e. SDN, using SDN technology we can choose routing path to transmit packets through network. This framework is easy to learn and use.

II. DESIGN

1. Architecture

CLOUDWATCHER is nothing but an administrator who monitors overall network and tries to ensure each packet passes through existing security device. Basically architecture involves three components:

I) Device and policy manager,
II) Routing rule generator,
III) Low rule enforcer.

Overall architecture is shown in dig.

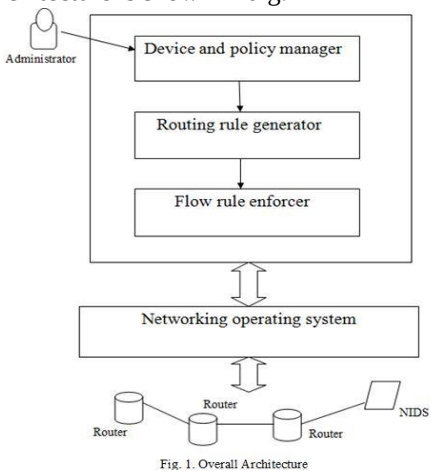


Fig. 1. Overall Architecture

2. Working

Before using security device through CLOUDWATCHER, first we need to register them. This can be done using basic information about security devices such as:

I) Security device id: this acts as unique identifier,
II) Device type: which indicates function perform by device,
III) Location: this shows where we actually install device,
IV) Installation mode: this show how the device is installed that is either passive or in-line mode,
V) Additional function: which indicate how and what security measure supported by device.

I) Security Policy

To provide secure service related to network monitoring cloud administrator must define certain policy which must contains following fields:

To control flow administrator has to define various conditions, SDN specifications supports matching fields on which these condition depends. For example in open flow specification the admin uses many matching fields, which help to setup a condition field. Admin can uses three matching field to set up flow condition field to perform this work we consider four tuple information, this tuple include,

1. Source IP address
2. Destination IP address
3. Source IP port
4. Destination IP port

Administrator specifies in device set field that which security device involves two monitor network packets.

II) Control Network Flow

Administrator satisfies security requirement by routing this packet based on the flow condition policy. While passing packets through specific routes it should check following conditions:

1. Network packets should pass through node which attached to security device.
2. Administrator ensures that network packet must go through shortest path between source and destination.

To find shortest path between source & destination node there are different routing algorithms, but we cannot directly use these algorithms, because these existing algorithms not able to find out where exactly security devices are placed.

Newly updated technology such as open flow allows us to control network flow according to location of security device [3]. Using this technology we propose to algorithms.

I) Shortest Through
II) Shortest Inline

A network consist of two or more nodes (Host, Routers, and Switches) and these nodes are link by physical link. In these networks we need to find shortest path between start & end node for efficient delivery of packet. We assume m_{ij} which shows delivery of data from node i to node j, l_i it is amount of data provided by node i , n_{ij} is optimal distance of path i and j. We considered distance between each two nodes as one.

$$\min \sum n_{i,j} m_{i,j}$$

$$\text{s.t} \sum_{j=1}^{n} m_{i,j} - \sum_{k=1}^{n} m_{k,i} = l_i \quad \text{for i=1,2,....n} \quad (1)$$

$$m_{i,j} >= 0 \quad \text{for i,j=1,2,...n}$$

From above equations we find minimum path between start and end node. Using result of minimum path we define certain policy base on which data will be send through particular path between source & destination. For explaining our algorithms we define certain parameters as follows

1. Start node: - A node which send packet.
2. End node: - A node which receives packets.
3. Security link: - It is link where security devices are installed.
4. Security node: - Anode to which security device is attach.

Algorithm 1 (Shortest through): These algorithms monitor network packets passively.
Algorithm 2 (Shortest inline):- This algorithm uses inline security device for network prevention system.

To explain a proposed algorithms more clearly we provide following example.

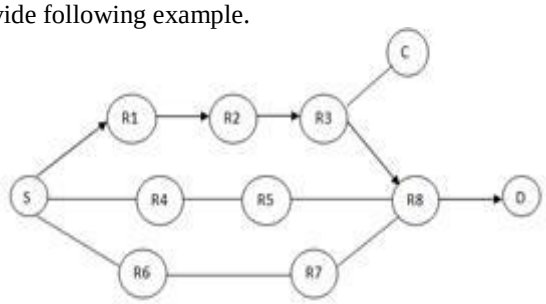


Fig 2. Algorithm 1

In above diagram we used 8 routers R1,R2,R3,R4,R5,R6,R7,R8,S & D. where S is start node and D is destination node. After applying minimum cost algorithm we find a minimum distance between S-R6-R7-R8 -D is minimum, but it doesn't contain security device so we transmit data through S-R1-R2-R3-R8-D which contains security nod R3.

Algorithm 2(shortest through):

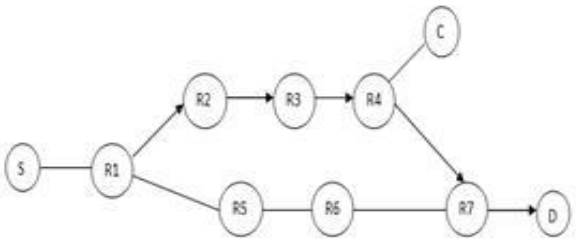


Fig 3. Algorithm 2 (shortest through)

In this algorithm we need to find all possible paths from start node to end node. We must pass packet through each intermediate security node. Reject those paths which doesn't contain security node. Finding this path is more difficult as compare to finding minimal path between nodes

because we should take care of all intermediate nodes. This algorithm uses open flow technique which includes following parameters such as Flow table which consist of flow entries like Match field-to match against packet consist of the ingress port, packet headers & optionally metadata. Counters-to update for matching fields. Instructions- to modify the action set or pipeline processing.

Algorithm 2(shortest inline):-

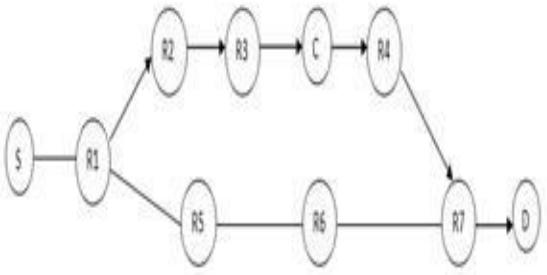


Fig 4. Algorithm 2(shortest inline)

Previously, we presented an algorithm in which network monitor by security device in passive mode. In this algorithm security device is commissioned as an inline mode. For uninvolved checking units, we can basically uncover a way passing through each security node; then again, in this case there is some security device, which works in the active mode, we must think about both of security nodes and security links. In spite of the fact that a way incorporates two nodes for a connection, it doesn't ensure that the connection is utilized for the way, since every node could be connected to alternate nodes. Accordingly, Algorithm 2 has a routine checking if security connections are incorporated or not.

III. IMPLEMENTATION

To crosscheck our ideas we can use open flow specification to implement proposed framework [3]. We explain our proposed framework in following manner:

1) *Device and policy manager*:-This componentconsists of two tables.

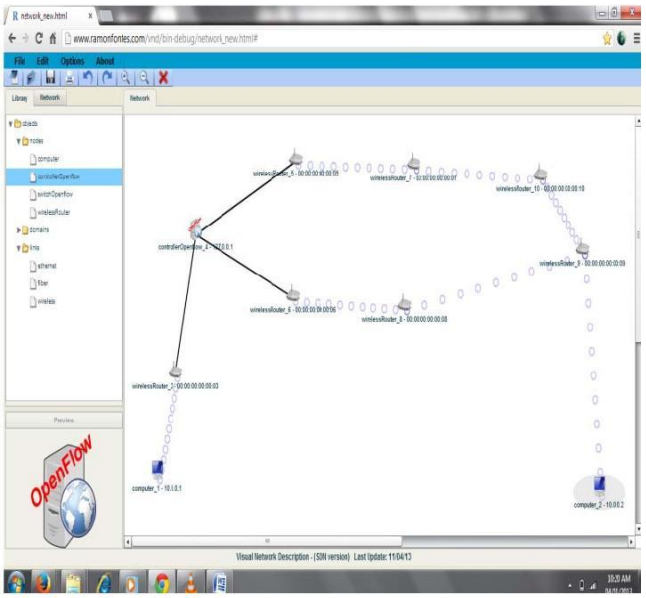
   Device Table: This table holds the information regarding the security device.

   Policy Table: This holds information regarding security policy.

These tables retrieve using hash function.

2) *Routing rule generator*: The topology of theunderline network need to understand by this by model. All information from routers & switches collected by this model and represented it in graph structure. To estimate cost of the network link, this module needs the information of network status. This module finds out the shortest path between nodes using Dijkstra's algorithm [7].
3) *Flow rule enforcer*: This module uses a routingrule strategy and use them to control the flow for open flow router and switches.

IV.EVALUATION



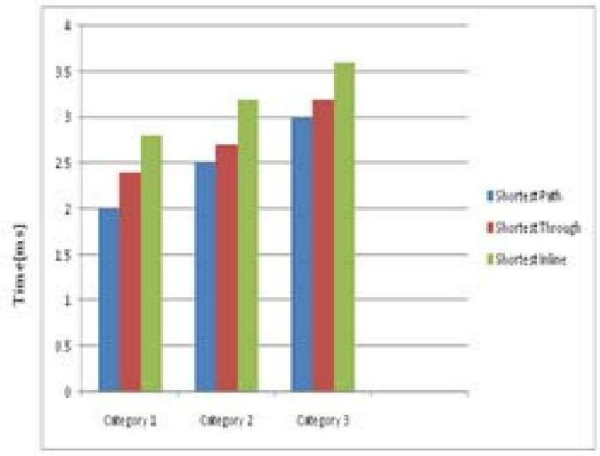


Fig 5. Flow Rule Generation Time Measurement

V. RESULTANDDISCUSSION

Our system has some drawbacks. In some cases CLOUDWATCHER not able to create routing paths. Consider an example, if system admin defines two security devices can not correspond with one another, then our algorithm will not be able to find any paths. For this at least network administrator gives warning by showing messages for such difficult routing path. Assuming that there are many new streams in a cloud system, because of this network performance may be affected. This problem can solved by implying CLOUDWATCHER on distributed system.

VI. CONCLUSION

Cloud watcher is another schema to help a cloud admin screen a cloud arranges effectively and productively, furthermore it furnishes security overseeing as an administration to its occupants. The proposed steering calculations have the capacity to furnish dynamic overseeing of system streams in cloud systems in a streamlined manner. All necessary functions and operations

can be accomplished by a basic script dialect, we accept that Cloud watcher can furnish handy and practical system security overseeing in a cloud system.

REFERENCES

[1] Amazon data center size. http://huanliu.wordpress.com/2012/03//amazon-data-center-size
[2] OpenFlow Specification v1.1.0. http://www.openflow.org/documents/openflow-spec-v1.1.0.pdf.
[3] AsiaCloudForum. Cloud security attacks – are public clouds at risk? http://www.asiacloudforum.com/content/cloud-securityattacks-are-public-clouds-risk.
[4] Natasha Gude, TeemuKoponen, Justin Pettit, Ben Pfaff, Martin Casado, Nick McKeown, and Scott Shenker. Nox: towards an operating system for networks. July 2008.
[5] IETF.OSPF. http://tools.ietf.org/html/rfc2328..
[6] David G. Luenberger and Yinyu Ye. Transportation and Network Flow Problems. In *Linear and Nonlinear Programming*,November 2010.
[7] Moshe Sniedovich. Dijkstra's algorithm revisited: the dynamic programming connation. In *Control and Cybernetics Journal*, 2006.