Design and Implementation of Cloud Security Defense System with Software Defined Networking Technologies

Sin-Fu Lai
Dept. of Communications Engineering
National Chung Cheng University
Chiayi County, Taiwan
michellchasel@gmail.com

Hui-Kai Su*
Dept. of Electrical Engineering
National Formosa University
Yunlin County, Taiwan
hksu@nfu.edu.tw

Wen-Hsu Hsiao
Dept. of Applied Digital Media
Wu-Feng University
Chiayi County, Taiwan
shianws@gmail.com

Kim-Joan Chen
Dept. of Electrical Engineering
National Chung Cheng University
Chiayi County, Taiwan
ieekjc@ccu.edu.tw

***Abstract*—In recent years, the rapid development of cloud computing and software-defined networking. Cloud security issues are more important, we hope to use the virtualization technology of cloud to implement vIDS and vFirewall so that we can according to the requirements of tenants dynamically adjust and replace hardware equipment with software to cut down the high cost. This paper proposes and we design Security Policy Decision System (SPDS) in the cloud, by using OpenStack to create multiple vIDS to detect the distributed denial of service (DDoS) attacks and combined with multiple vFirewall to filter the attack packets so that we can introduced SDN technology to distribute the flow to multiple vIDS to analyze attack packets or direct traffic elsewhere do processing. We hope to take advantage of SDN in the cloud through vIDS, SPDS (security policy decision system) and vFirewall to effectively alleviate the impact of the DDoS attack in the cloud environment.**

Keywords—software-defined networking; vIDS; vFirewall; Security Policy Decision System (SPDS); Distributed denial of service (DDoS)

I. INTRODUCTION

With the rapid development of Internet technologies and cloud computing, there are various network attacks increasing year by year. Traditional IDS and Firewall are not only expensive but also unable to meet the huge computing requirement of cloud. We want to use SDN and cloud virtualization technology not only can let hardware IDS and Firewall virtualize but also we can according to the requirements of the tenants dynamically adjust the resources and decrease the cost of buying the hardware. However, the main security issues in the cloud are botnet and distributed denial of service attack. Botnet can through the e-mail, instant messaging software (eg, MSN or IRC: Internet relay chat) or vulnerabilities of the computer system to intrude your computer, and then hiding in the program. Botnet viruses and Trojans are similar, but the Trojans only attack specific targets, less likely to be implanted by Trojan mainframe and try to attack another computer. Comparing to traditional networks, hackers can easily to use features of the botnets and virtualization technologies to consist of computer network to launch a large scale network attacks from outside the cloud or according to internal host's vulnerability to intrude to attack and destroy, so Botnet and DDoS attack will cause the depletion of resources and the cloud service environment network paralysis. So we need to achieve good detection, monitoring and policy management and other protective measures to prevent attack in the cloud.

II. RELATED WORK

A. OpenStack Introduction

OpenStack is a free and open-source software platform for cloud computing, mostly deployed as an infrastructure-as-a-service (IaaS). OpenStack began in 2010 as a joint project of Rackspace Hosting and NASA [1].

OpenStack is a cloud operating system that controls large pools of compute, storage, and networking resources throughout a datacenter, all managed through a dashboard that gives administrators control while empowering their users to provision resources through a web interface [2].

OpenStack project contains four internal modules, Dashboard module, Compute module, Networking module, Storage module.

Dashboard module centralized management of the other three modules, consisting of a set of OpenStack cloud resource sharing as shown in Fig. 1.

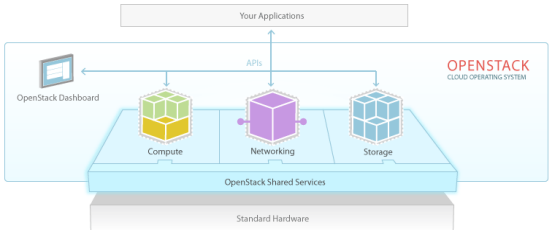

Fig. 1. OpenStack module[2]

B. *IDS (Intrusion Detection Systems)*

IDS (Intrusion Detection System, IDS) is a monitoring tool by analyzing network packets or host system logs and compare existing attack signatures according to the rules, real-time detection, response and reply of abnormal behavior. Common tools such as Snort [3] is a lightweight Intrusion Detection System, and is cross-platform and free software, its main features: Snort is a Network-based intrusion detection system, using the Misuse to detect the malicious behavior

TABLE I. DETECTION METHODS

	Misuse (signatures)	**Anomaly**
Method	See whether the traffic has known signs of intrusion (exploit vs. vulnerability signatures).	See whether anomaly occurs in traffic.
Pros	Can find known intrusion efficiently.	Can detect unknown or complicated attacks.
Cons	Ineffective for new attacks (false negative). Can be easily evaded.	Possibly many false positives.

C. *Software-defined networking (SDN)*

Software-Defined Networking (SDN) is a network paradigm usually characterized by three fundamental aspects:

• A clear separation of network forwarding and control planes.

• The abstraction of the network logic from hardware implementation into software.

• The presence of a network controller that coordinates the forwarding decisions of network devices.

Peer Hasselmeyer[4] mention that SDN separate and abstract elements in computer networks and provide "Network API" or "Network OS" to application programmers so that we can use controller centralized control and network programmable to improve network reliability and security, dynamic management, unified policy enforcement, reducing configuration errors, allowing network in the planning and management more simple and convenient as shown in Fig. 2.

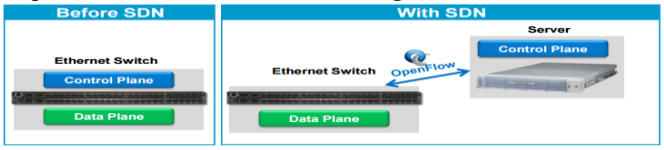

Fig. 2. Before SDN and With SDN [4]

Sakir Sezer et al.[5] mention SDN architecture mainly divided into three layers, four interfaces as shown in Figure IV

• Three Layers:

(a)**Infrastructure Layer** Include Network Node such as router, switch, virtual switch

(b)**Control Layer** Controller through Southbound API and Northbound API to set command dynamically manage, modify flow entries and communicate with network devices to achieve effective network monitoring.

(c)**Application Layer:** Develop related application such as Traffic/security monitoring.

Sakir Sezer et al.[5] also mention the operation of SDN (switch-controller) as shown in figure

(Step1) First packet of a new flow arrives at the switch from the sender.

(Step2) Switch checks for a flow rule for this packet in the SDN cache. If a matching entry is found, the instructions associated with the specific flow entry are executed (e.g., update counter, packet/match fields, action set, metadata).

(Step3) If no match is found in the flow table, the packet may be forwarded to the controller over a secure channel (step 3). Using the southbound API (e.g., OpenFlow, ForCES, PCEP), the controller can add, update, and delete flow entries, both reactively (in response to packets) and proactively.

(Step4) The controller executes the routing algorithm, and adds a new forwarding entry to the flow table in the switch and to each of the relevant switches along the flow path.

(Step5) The switch then forwards the packet to the appropriate port to send the packet to the receiver.

D. *OpenFlow*

OpenFlow[6] is a SDN protocol that proposed by Stanford university in 2008 through the OpenFlow technologies network planning staff or network managers can add or delete the flow entries into flow tables to manage the network. However, OpenFlow standard is defined by Open Networking Foundation [7]. Today, OpenFlow also support many products[8].We can see that the SDN technologies and related research are popular and increasing development.

Controller through the security channel connecting to each Openflow Switch as shown in Fig. 3.

Three elements in OpenFlow：

(1) **Flow Table:** Defined the action of each flow to notify switch how to process each flow.

(2) **Secure Channel: Connect switch and remote switch to set command and sent the packets.**

(3) **OpenFlow Protocol:** Provide a standard let controller can communicate with switch.

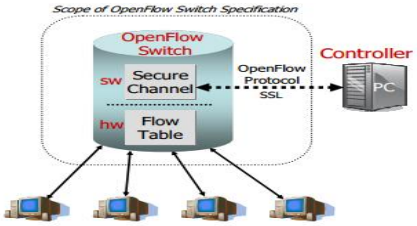

Fig. 3. Idealized OpenFlow Switch. The Flow Table is controlled by a remote controller via the Secure Channel. [6]

Main Components in an Open vSwitch as shown in Fig. 4[9]:

(a) One or more flow tables and a group table for packet lookups and forwarding.

(b) One or more OpenFlow channels to an external controller.

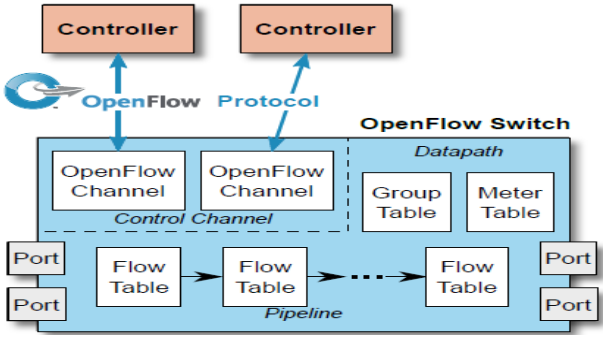

Fig. 4. Main Components in an OpenvSwitch [9]

Diego Kreutz et al.[10] mention that in an SDN/OpenFlow architecture, there are two main elements, the controllers and the forwarding devices, as shown in Fig. 5.

An OpenFlow enabled forwarding device is based on a

pipeline of flow tables where each entry of a flow table has three parts:

Inside an OpenFlow device, a path through a sequence of flow tables defines how packets should be handled. When a new packet arrives, the lookup process starts in the first table and ends either with a match in one of the tables of the pipeline or with a miss (when no rule is found for that packet). A flow rule can be defined by combining different matching fields, as illustrated in Fig. 7.

If there is no default rule, the packet will be discarded. However, the common case is to install a default rule which tells the switch to send the packet to the controller (or to the normal non-OpenFlow pipeline of the switch).

Possible actions include:
(a) Forward the packet to outgoing port(s);
(b) Encapsulate it and forward it to the controller;
(c) Drop it;
(d) Send it to the normal processing pipeline; and
(e) Send it to the next flow table or to special tables, such as group or metering tables introduced in the latest OpenFlow protocol.

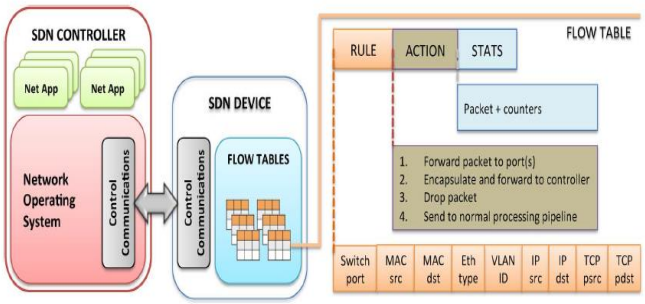

Fig. 5. OpenFlow-enabled SDN devices.[10]

TABLE II. : OPENFLOW CONTROLLER[15]

Name	Platform	Organization	Features	Site
NOX/POX	C++/Python	Stanford	the first OpenFlow controller	http://www.noxrepo.org/
Beacon	Java	Stanford	support both event-based and threaded operation, provide Web UI	https://openflow.stanford.edu/display/Beacon/Home
Floodlight	Java	Big Switch	developer-friendly with plenty documents	http://floodlight.openflowhub.org/
Ryu	Python	NTT laboratories OSRG group	with OpenStack support	http://osrg.github.io/ryu/
Maestro	Java	Rice University	Multi-threaded support	http://code.google.com/p/maestro-platform/
OMNI	Pyhton/Java	Universidade Federal do Rio de Janeiro	web interface on NOX	http://www.gta.ufrj.br/omni/
McNettle	Haskell	Yale University	high-level declarative expressive language, multi-core optimization	http://haskell.cs.yale.edu/nettle/mcnettle/
Trema	Ruby/C		virtual network DSL	http://trema.github.io/trema/
OpenDaylight Controller	Java	Linux Foundation	provide REST API and web GUI	http://www.opendaylight.org/

Wenfeng Xia et al.[11] list OpenFlow Controller as shown in Table 1. Today the mainly popular controller is Ryu and OpenDaylight Controller, Ryu is written by Python that can control the OpenvSwitch through the controller. Network planning staff or network manager also can use Mininet to plan and design the network topology to collocate Ryu OpenFlow Controller and OpenStack.

Ryu Controller also support OpenStack API let network managers dynamically allocate the resources in the cloud as shown in Fig. 6.

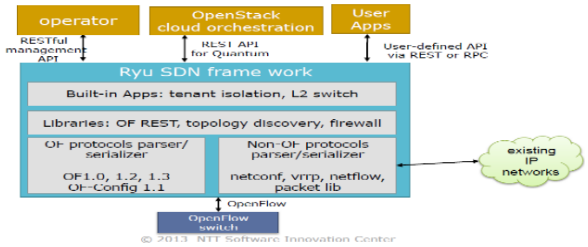

Fig. 6. Ryu Controller Architecture [11]

III. SYSTEN ARCHITECTURE AND DESIGN

The main goal of the cloud defense system will protect each VM will not attack by external and internal attack.

So we reference [12] the DDoS attack protection mechanism in SDN and choose the network-based mechanisms traffic analysis and dynamic rules updating to design the cloud defense system.

In the cloud, by using the SDN to plan and design the cloud security control system as shown in Fig. 7. Through the Open vSwitch splitting the flow to multiple vIDS to detect and analyze the attack packets so that we can according to the threshold to judge abnormal behavior. However, we can combine with vFiewall according to the requirements of tenants to update the firewall polices and through SPDS with OpenFlow Controller doing security control and making decision to achieve the security protection in the cloud.

A. System Architecture

We choose OpenStack to build the system architecture in the cloud environment and spilt many security perimeters to divide into many departments in the company, each VM inside the department will regard as a tenant in the cloud. However, we consider the flexibility of users by using the Open vSwitch to distribute the flow and according to the requirements of tenants dynamically adjust through multiple vIDS to detect attack packets or redirect the flow to vFirewall filter the packets.

Security Management and Policies: Divided into external and internal cloud protection

• **Defense outside the cloud:** When the malicious attacks into the cloud, we will use Open vSwitch to distribute flow to multiple vIDS to analyze the packets. Once detect the malicious the attack flow, we will use Open vSwitch to distribute the flow to multiple vIDS according to the threshold to check the packets whether are abnormal so that through the vIDS alerting to the SPDS and SPDS will choose appropriate the defense policies to store in the defense event database. SPDS will update the defense policies to the vFirewall to filter the packets and achieve the protection outside the cloud.

• **Defense inside the cloud:** When we found the VMs attack by the botnet destroying in the cloud, the SDN Ryu controller will communicate with the SDPS to match the defense rules according to the rules to add the entries to switch flow table to block the attack in the cloud to achieve the internal cloud protection.

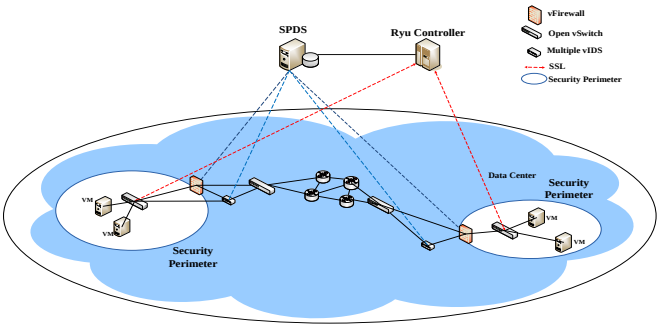

Fig. 7. Cloud Security Defense System architecture

B. System Design

When the network attacks launch, the two components are monitoring defense system and security policies desicsion system in the cloud will interact with each other as shown in Fig 8. If the monitoring modules detect the suspicious attacks,it will store the attack information into the statistics database when the attack information over the threshold and the Snort will send alert information. The alert inforamation consist of attack information such as attack types, attack source port, attack destination port, attack protocol so that when the information written and start analyzing to trigger the security policies desicision system to dymaically achieve protection in the cloud.

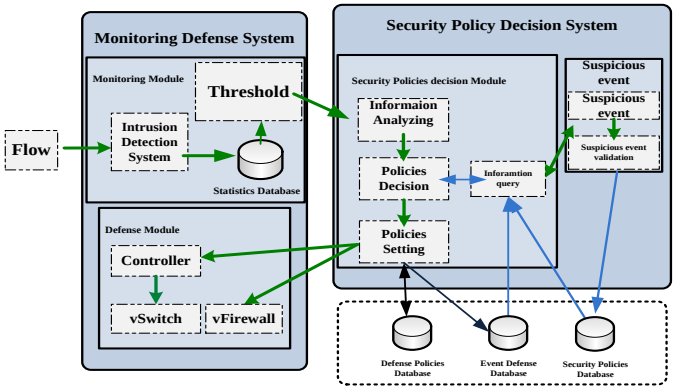

Fig. 8. System modules

We choose the Snort to detect the attack packets and the alert information as shown in TABLE III.

TABLE III. ALERT INFORMATION

ID	Protocol	S_IP/Port	D_IP/Port	Attack_Name	Attack_type	Alert_type
1	UDP	67.195.113.227:2010	140.123.107.21:80	UDP Flood	DoS	Confirmed
1	UDP	140.21.112.2:3333	140.123.107.60:1010	ET TROJAN IMDDOS Botnet	Trojan	UnConfirmed
2	ICMP	68.121.15.20:3300	140.123.40.10:1100	BACKDOOR neurotickat1.3	Backdoor	Confirmed

C. Designing SDN distributed the flow mechanism

vIDS architecture design as shown in Fig. 9. vIDS compose of one or more VMs through Open vSwitch processing the packets redirection and each VM will allocate at least one detector to support the function of the monitor.

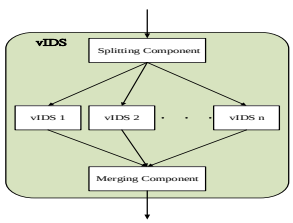

Fig. 9. Architecture of the internal vIDS

The architecture of the vIDS distributing the flow as shown in Fig. 10 through OpenFlow distributing the flow and detect attacks by multiple vIDS can effectively reduce the loading of the virtual devices to achieve the goal of the load balancing.

We consider the many factors e.g. cpu, memory, input/output to determine the resources of the detector(VMs) whether should be allocated and released. However, it is hard for comparing with different resources requirements of the VMs so we consider the resources requirements of the single VM. We reference the equation1 based on the cpu, memory, net represented the resources of the VMs utilization. If moe resources will extensive use, the volume will be larger.

$$volume = \frac{1}{1-cpu} * \frac{1}{1-mem} * \frac{1}{1-net} \quad (1)$$

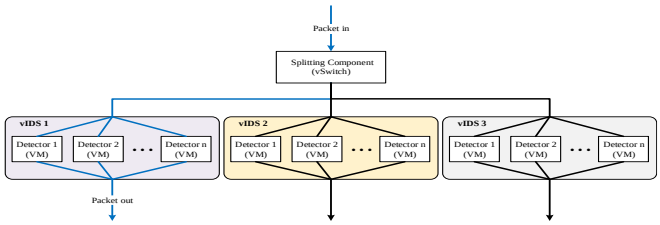

Fig. 10. Architecture of the vIDS distributing the flow

In the paper, we use the Vyatta[13] as the operating system of the vFirewall throuth Vyatta can provide many functions of the the advanced routing, firewall, RIP, OSPF v2 / v3, BGP, IPv6, IPSec, SSL VPN, SNMP, NAT and supporting the software-defined networking.

Vyatta is a software of the router/firewall, the architecture supports the standard of the x86 hardware so that x86 host just install Vyatta will be an enterprise router / firewall network equipment. The vFirewall as shown in Fig. 11 and the as shown in TABLE IV.

Fig. 11. Vyatta VC6.6 R1 operating system as vFirewall

TABLE IV. DEFENSE RULE SAMPLE

	Protocol	Source Address	Source Port	Destination Address	Destination Port	Action
1.	TCP	140.123.10.2	any	140.123.107.21	3311	deny
2.	TCP	140.21.104.2	any	140.123.107.21	1010	deny
3.	TCP	68.121.15.20	any	140.123.40.10	1100	deny

D. System Processes

When the network attacks launch to attack the cloud security defense system and the components will interact and achieve the cloud security protection and attack simulation scenarios as shown in Fig. 12.

Simulation scenarios are divided into external and internal attack:

• **Attacks protection outside the cloud:** Simulate the attacks launch attack outside the cloud

① Attackers install hping3 launch TCP SYN flood or ICMP attacks

② Open vSwitch distribute flow to multiple vIDS to detect attack

③ vIDS according to the attack signatures detect whether the packets are malicious attack

④ vIDS detect the abnormal packets will sent alert information to SPDS

⑤ SPDS determine appropriate defense policy transfer policy and store in the defense event database

⑥ SPDS update vFirewall defense rules and filter the attack packets

• **Attacks protection inside the cloud:** Simulate the attacks launch attack inside the cloud

Simulate the VMs attack other VMs in the cloud

⑦ VM1 and VM2 launch TCP SYN flood or ICMP attacks

⑧ Ryu Controller communicate with SPDS and match the defense rule finally reply to Ryu Controller

⑨ According the rule add entries into Switch Flow Table

⑩ Block the attack packets

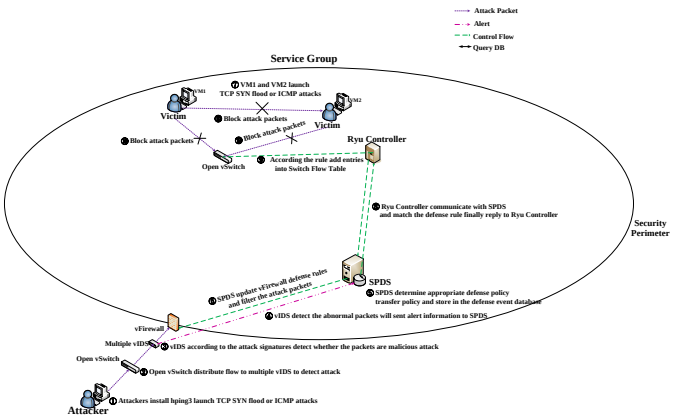

Fig. 12. Attack simulation scenarios (step by step)

Flow chart as shown in Fig. 13 and and attacks sequence diagram in Fig. 14 When the monitoring module detect the attacks over the threshold confirming the attack behaviors, Snort will send alert information based on the attack message to analyze and according to the event defense database and cloud database to choose the appropriate the security policies to transfer to rules into database and update to the vFirewall to achieve the external cloud protection. However, when VMs attack each other in the cloud, SPDS will communicate with Ryu controller to match the defense rules and add the entries to Open vSwitch flow table to achieve the internal protection.

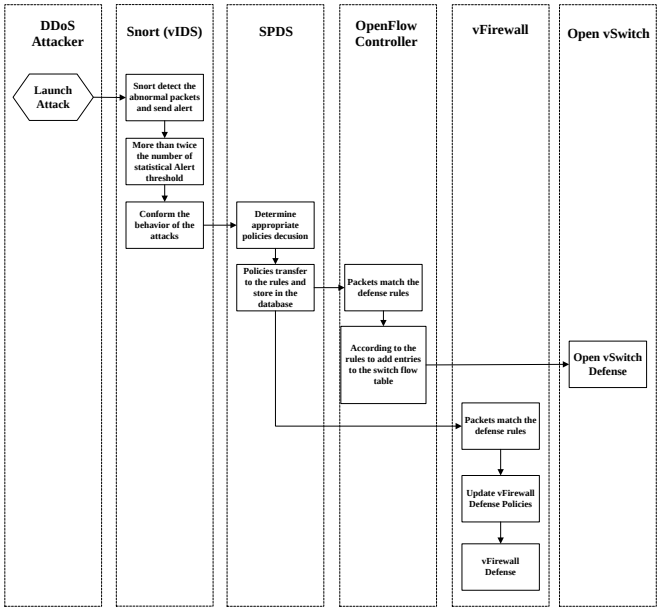

Fig. 13. Attacks flow chart

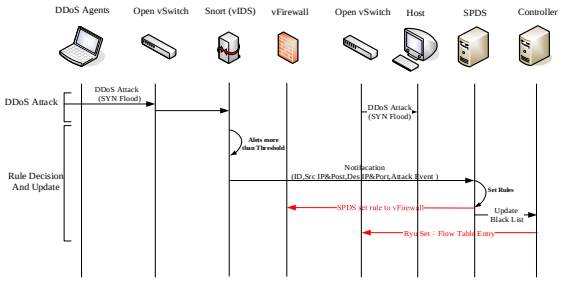

Fig. 14. Attacks Sequence diagram

When the program initialize completely, SPDS will standby and wait for analyzing the files written. When the information written, SPDS will operate and generate the appropriate defense rules. Security policy decision system process as shown in Fig. 15.

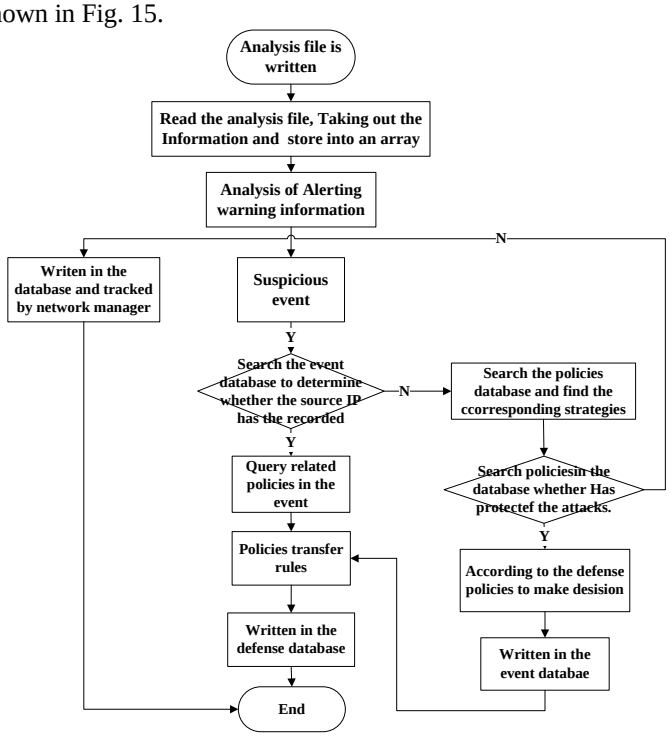

Fig. 15. Security Policy Decision System Process

IV. EVALUATION DATA

We choose the hing3[14] tool and try to launch ICMP and TCP SYN Flood attack internal and external cloud security defense system in the future to evaluate performance of the vIDS and vFirewall

• **vIDS performance evaluation**: According to the equation5 based on a single vIDS can detect the maximum packets loading to add new VM to achieve the load balancing of vIDS analyzing the packets.

• **vFirewall performance evaluation**: Setting the rules and filter the packets and according to a single VM can process the maximum packets loading to add a new VM.

V. CONCLUSION AND FUTURE WORK

In this paper, we will transform traditional hardware IDS and Firewall architecture into the cloud environments and collocate OenStack can according to the requirements of the tenants add or delete VMs. By using the SDN technologies to distribute the flow and combined with the intrusion detection systems analyzing the packets whether are abnormal.

Designing the SPDS in the cloud environments to implement the security detection and defense architecture through query the database and set the defense rules to update the vFirewall and Open vSwitch to achieve the good detection and protection. However, in order to response to the multiple attacks only use the vIDS and vFirewall that are not enough. So we want to introduce SDN technologies to distribute the flow to multiple vIDS to detect and analyze whether packets are abnormal.

But we consider the better analysis process through the public cloud strongly computing resources combining with the network behavior analysis system to analyze more attacks. We believe that the cloud defense system can collect the signatures of the attacks and classify similar attack signatures according to the different types of attacks to choose the appropriate policies of the network allocation in the future.

[1] OpenStack introduction, https://en.wikipedia.org/wiki/OpenStack

[2] OpenStack, https://www.openstack.org

[3] Snort, http://www.snort.org

[4] OFELIA presentation, http://www.fp7-ofelia.eu/assets/Publications-and-Presentations/SDN-The-Next-Wave-of-Networking.pdf

[5] B. Fraser, D. Lake, C. Systems, J. Finnegan, N. Viljoen, and S. O. E. N. Etworking, “Are We Ready for SDN ? Implementation Challenges for Software-Defined Networks,” Communications Magazine, IEEE, vol. 51, pp. 36-43, 2013.

[6] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, “Openflow : Enabling Innovation in Campus Networks”,ACMSIGCOMM Computer Communication Review, Vo1 38, pp.69-74,April 2008.

[7] Open Networking Foundation, https://www.opennetworking.org/about/onf-overview

[8] 6WINDGate Networking Software, https://www.opennetworking.org/sdn-openflow-products

[9] Network Foundation, “OpenFlow Switch Specification: Version 1.5.0,” Dec. 2014.

[10] D. Kreutz, F. Ramos, P. Verissimo, C. Rothenberg, S. Azodolmolky, and S. Uhlig,“Software-Defined Networking: A Comprehensive Survey,” Proceedings of the IEEE, vol. 103, no. 1, pp. 14-76, Jan. 2015.

[11] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie, “A Survey on Software-Defined Networking,” Communications Surveys & Tutorials, IEEE. vol. 17, 2015, pp. 27-51.

[12] Q. Yan, R. F. Yu, Q. Gong, and J. Li, “Software-defined networking (SDN) and distributed denial of service (DDoS) attacks in cloud computing environments: A survey, some research issues, and challenges,” Communications Surveys & Tutorials, vol. 18, 2016, pp. 602-622.

[13] Vyatta, https://en.wikipedia.org/wiki/Vyatta

[14] hping3, http://tools.kali.org/information-gathering/hping3