A Filter Tree Approach to Protect Cloud Computing against XML DDoS and HTTP DDoS Attack

Tarun Karnwal, Sivakumar Thandapanii, and Aghila Gnanasekaran

Abstract. Cloud computing is an internet based pay as use service which provides three type of layered services (Software as a Service, Platform as a Service and Infrastructure as a Service) to its consumer on demand. These on demand service facilities is being provide by cloud to its consumers in multitenant environment but as facility increases complexity and security problems also increase. Here all the resources are at one place in data centers. Cloud uses public and private APIs (Application Programming Interface) to provide services to its consumer in multitenant environment. In this environment Distributed Denial of Service attack (DDoS), especially HTTP, XML or REST based DDoS attacks may be very dangerous and may provide very harmful effects for availability of services and all consumers may get affected at the same time. One other reason is that because the cloud computing users make their request in XML and then send this request using HTTP protocol and build their system interface with REST protocol (such as Amazon EC2 or Microsoft Azure) hence XML attack more vulnerable. So the threaten coming from distributed REST attacks are more and easy to implement by the attacker, but to security expert very difficult to resolve. So to resolve these attacks this paper introduces a comber approach for security services called filtering tree. This filtering tree has five filters to detect and resolve XML and HTTP DDoS attack.

Keywords: Economical Distributed Denial of Service (EDDoS), Militant environment, Distributed Denial of Service(DDoS) Attacks, Pay as Use, Cloud Security, SaaS, Paas, IaaS.

1 Introduction

Cloud computing is a combination of distributed system, utility computing and grid computing. Cloud Computing uses combination of all these three in

Tarun Karnwal · Sivakumar Thandapanii · Aghila Gnanasekaran
Dept. of Computer Science, Pondicherry University
Puducherry, India
e-mail: {karnwals,Aghila}@gmail.com,
tsivakumar.csc@pondiuni.edu.in

A. Abraham and S.M. Thampi (Eds.): Intelligent Informatics, AISC 182, pp. 459–469.
springerlink.com © Springer-Verlag Berlin Heidelberg 2013

virtualized manner. Cloud computing converts desktop computing into service based computing using server cluster and huge databases at data center. Cloud computing gives advanced facility like on demand, pay per use, dynamically scalable and efficient provisioning of resources. Cloud computing the new emerged technology of distributed computing systems changed the phase of entire business over internet and set a new trend. The dream of Software as a Service becomes true; Cloud offers Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS). Cloud offers these services with the help of Web Services.

Cloud computing providing services to its consumers at abstract level and take care of all the internal complex tasks. With cloud computing consumer life became easy. But "as the nature rule with increase in facility vulnerability also increases".

Similarly Cloud provides the facility to consumers in the same way it provides facility to attackers also. There are more chance of attacks in cloud computing. As cloud computing mainly provides three types of services so in each layer have some soft corners which invite attackers to attack. Some of these soft corners are (1) SaaS vulnerability as Insecure Application Programming Interface (API), Account or Service hacking, Attack on cloud firewall / Attack on public firewall, Attack on consumer browser, Integrity, Confidentiality and Availability (2) PaaS vulnerability as Insecure Application Programming Interface (API), Unknown risk profile (Heartland Data Breach), Integrity, Confidentiality and Availability (3)IaaS vulnerability as Data leakage in Virtual Machine, Shared technology issues, Integrity, Confidentiality and Availability

So among all these different vulnerabilities Availability affects all three layers and more harmful. Every Cloud has its own APIs or adapters that need to be installed or consumed if anyone wants to use that Cloud. These adapters are publically available and this paper objective is to provide security to this Open API from HTTP and XML based Denial of service attacks.

The largest DDoS attacks have now grown to 40 gigabit barrier this year and may reach to 100 gigabits soon. So if someone threatens to bring down the cloud system with DDoS attack cloud may become worrisome. XML-based DDoS and HTTP-based DDoS are more destructive than the traditional DDoS because of these protocols widely used in cloud computing and lack of the real defense against them. HTTP and XML are important elements of cloud computing so security become crucial to safeguard the healthy development of cloud platforms. But as a virtual environment, cloud poses new security threats that differ from attacks on physical system.

2 Related Work

As Cloud Computing is new research area so security in cloud computing is also a very new and open challenge. Lot of research is going in security aspects in cloud computing. There are various latest real time examples in which cloud is suffering from new attacks among them HTTP and XML DDoS attacks are more common.

Since cloud computing security follows the idea of cloud computing, there are two main areas that security experts look at security in a cloud system: These are VM (Virtual Machine) vulnerabilities and message Availability between cloud systems. IaaS layer is more vulnerable as in [5] Shared Technology issues work on IaaS layer.

In [6] Data Loss or Data Leakage is a big problem on IaaS layer. There are many ways to compromise data deletion and alteration of records without a back-up of original content is an obvious example.

In [7] insecure API is big threat in cloud computing. Cloud computing providers exploit a set of software APIs that customer use to manage and interact with cloud services.

Various solutions and techniques exist for detection and protection from HTTP flood attack and XML attack in Cloud Computing.

Chu-Hsing Lin et. al. [8] is using Semantic Web concept to find flooding attack by dividing attacks in three categories but this solution limited to identifying malicious browsing behaviors. Tuncer et. al. [9] is using fuzzy logic to find flooding attack. This solution will give more false positive results. Liming Lu et. al. [10] using Probabilistic Packet Marking for IP Traceback. This method is useful only when we already have attackers IP Address in traceback but in real time it is not possible. Suriadi et. al. using client puzzle but if each packet of client request will pass through client puzzle filter then this solution will face time bottleneck problem. Ashley Chonka et. al. [11] are using BPNN scheme to detect DDoS attack but this scheme work on expert system based approximate threshold value. Until attacker will not cross threshold value it is possible that attacker may attack. M.A. Rahaman et. al. [12] are using inline approach but the disadvantage of this method is that it is securing only some properties of SOAP message. Further N. Gruschka et. al. are introducing first real XML SOAP Message wrapping attack on Amazon EC2 services in 2008. Here attackers are changing XML tags and making vulnerability in SOAP Message request validation. By which any unauthorized user can access the services of Amazon's EC2.

3 System Architecture

The Proposed architecture is having five modules from Client to Cloud Provider. Client requests resources from Cloud Providers by using SOAP message. This

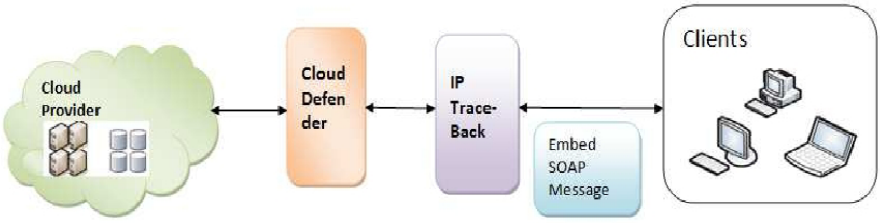

Fig. 1 Proposed Architecture Model

SOAP message has vulnerability of XML DDoS and HTTP DDoS attacks. So this paper introduces three modules which will provide security to SOAP message from these attacks. As a virtual environment, cloud poses new security threats that differ from attacks on physical system.

4 Embed Soap Message

Clients or Consumers use SOAP message to request any resource from cloud providers. SOAP message written in XML only because XML is universally acceptable language and it can run at any platform.

4.1 SOAP Signature

SOAP message is nothing but XML tags. The process of SOAP signature as: for every message part a reference element is created and the message part is hashed and cannibalized. The resulting digest added with digest value as well as the reference of signed message is added in URI field. In last this message part and digest cannibalized and put in Signed Info part and Signature element is added in security header.

4.2 Double Signature

To give the extra protection against XML rewriting attack Double Signature has been used by marking parameters (as number of children, number of header element and number of body element) in SOAP message and keeping these signed parameter in SOAP Header

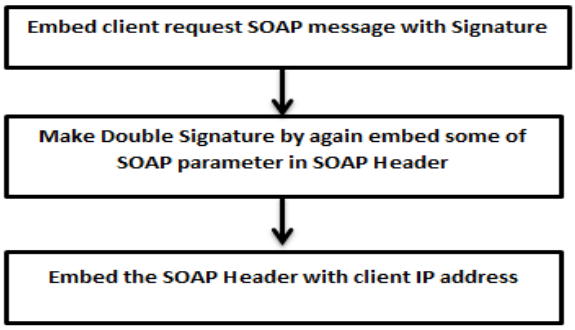

Fig. 2 Embed SOAP Message

4.3 IP Marking

SOAP message will mark at edge router using Flexible Deterministic Packet Marking scheme (FDPM). Three fields in the IP header are used for marking; they are Type of Service (TOS), Fragment ID, and Reserved Flag. A total of 25 bits (8 bit from TOS, 16 bit from TOS, 16 bit from identification field and 1 bit from offset field) are available for the storage of mark information if the protected network allows overwriting on TOS.

***Algorithm 1**: Embeded SOAP Message algorithm*

1. ***Input**: SOAP Message with XML tags, at router R, in network N*
2. ***Output**: client request*
3. *Requested SOAP message will embed with signature and header marked at router R*
4. ***if** wsse: Contained = true*
5. *computed_digest ← hash(wsse:QueryKey)*
6. ***else***
7. *computed_digest ← hash(wsse:SmallerValue, ec2:GreaterValue)*
8. ***for all** h ← wsse:Hash ∈ wsse:HashList*
9. *computed_digest ← hash(computed_digest, h)*
10. *replace(Data1.Value, computed_digest)*
11. ***for all** h ← wsse: count (children, header element, body element)*
12. *computed_digest ← hash(computed_digest, h)*
13. *replace(Data2.Value, computed_digest)*
14. ***return** Data1, Data2*
15. *set the bit array digest and mark to 0*
16. ***if** N does not utilize TOS*
17. *reserved_Flag:=0*
18. *7^{th} and 8^{th} bit of TOS:=0*
19. *length_of_Mark:=24*
20. ***else***
21. *reserved_Flag:=1*
22. ***if** N utilizes Differentiated Services Field*
23. *7^{th} and 8^{th} bit of TOS:=1*
24. *length_of_Mark:=16*
25. *else if h support Precedence but not priority*
26. *7^{th} bit of TOS = 1 and 8^{th} bit of TOS = 0*
27. *length_of_Mark:=19*
28. ***else if** N support Priority but not Precedence*
29. *7^{th} bit of TOS = 0 and 8^{th} bit of TOS = 1*
30. *length_of_mark:=19*
31. *decide the lengths of each part in the mark*
32. *digest:=hash(A)*
33. ***for** i=0 to k-1*
34. *mark[i].Digest:= Digest*
35. *mark[i].Segment_number:=i*
36. *mark[i].Address_bit:=A[i]*
37. ***for each** incoming p passing the encoding router*
38. *j:=random integer from 0 to k-1 // message divide in k bits*
39. *write Mark[j] into p.mark*

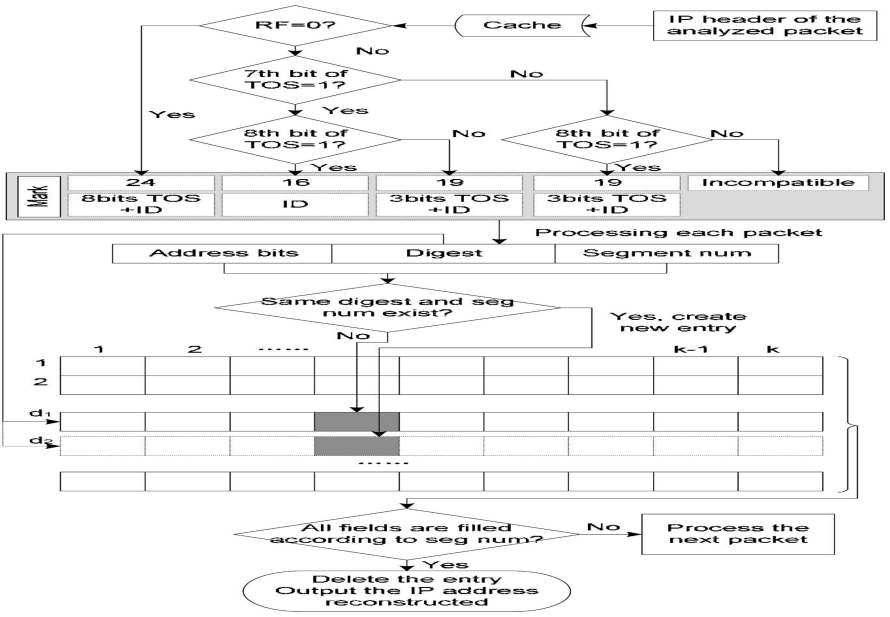

Fig. 3 Deterministic Packet marking

5 IP Trace-Back

IP Trace-Back is a logical file system. In proposed architecture IP Trace-Back stores vulnerable IP address provided by Cloud Defender. When client message request comes to IP Trace-Back, it matches coming message source IP address with already stored vulnerable IP address. If IP matched then it discard request message otherwise it send request message to Cloud Defender.

6 Cloud Defender

Cloud defender filters the attack in five stages. These five stages are

(1) Sensor Filter
(2) Hop Count Filter
(3) IP Frequency Divergence Filter
(4) Confirm legitimate user IP Filter
(5) Double Signature Filter

First four filters detect HTTP DDoS attack and fifth filter detects XML DDoS attack.

6.1 Detect Suspicious Message

6.1.1 Sensor

Sensor monitors the incoming request messages. If the sensor finds that there is hypothetical increase in the number of request messages coming from any particular consumer then it marks those messages as suspicious IP otherwise send to next filter.

6.1.2 HOP Count Filter

It will calculate the Hop Count value and compare with stored Hop Count value. If no match then it marks those messages as suspicious IP otherwise send to next filter.

6.1.3 IP Frequency Divergence

Because in DDoS attacker will not generate different request message every time so he has need to send same request messages again and again. If found same frequency of IP messages then it marks those messages as suspicious IP otherwise send to next filter.

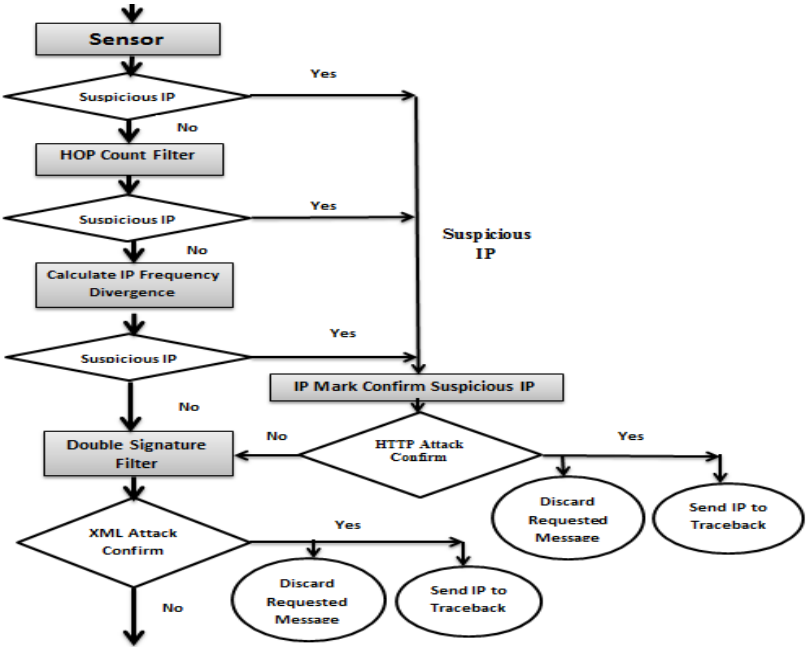

Fig. 4 Cloud Defender

6.2 Detect HTTP DDoS Attack

All suspicious packets come to the Puzzle Resolver. It resolves the SOAP header of these suspicious messages. Firstly it finds the suspicious messages IP addresses and then send the puzzles to these IP address If the suspense IP address send the correctly solved puzzle to puzzle resolver it means it is genuine client request otherwise puzzle resolver drops the request message and send suspicious IP address to IP Trace-Back otherwise it send the request message to Double signature filter.

6.3 Detect Coercive Parsing/XML DDoS Attack

Check the incoming request message for any open tag. If open tag found in incoming message then it discards that message otherwise send the request message to cloud provider to provide services to clients.

Because of cloud defender is working in multi-tenant environment. So in order to send multiple client requests to cloud defender we will limit the number of client request at a particular moment of time. Suppose coming client request are N and threshold is N1. If number of client requests are greater than N1 (here N>N1) then Cloud Defender will send these requests to different filters otherwise Cloud Defender will send the request packets directly to Double Signature Filter to check for XML DDoS attack.

In a DDoS attack, take place with zombies by a single attacker (master). These are nothing but zombie machines or attacker uses virtual machines and open thousands of tags by using these virtual machines and make DDoS attack on cloud provider. General attack traffic distribution will obey Poisson distribution approximately. The Poisson distribution function for DDoS attack traffic is shown below

$$P_k = \lambda^N * e^{-\lambda} / N! \quad (1)$$

Where λ is a positive real number, equal to the expected number of occurrences that occur during the given interval, and k is a non-negative integer, N=0, 1, 2,… In information theory, the information entropy is a measure of the uncertainty associated with a random variable.

Algorithm 2. *Cloud Defender algorithm*

1. ***Input****: N, N1, ∀N ∈[1, . . . , n], the final TTL Tf, the initial TTL Ti, stored hop-count in IP packet Hs, N is the length incoming request*
2. ***Input****: H_{fd}= frequency divergence.*
3. ***Output****: Legitimate client request*
4. ***If*** *N>N1 then*
5. ***for each*** *packet N compute the hop-count*
6. *Hc=Ti-Tf*
7. ***If*** *Hc!=Hs then*
8. *Continue*
9. ***else if***
10. *mark request packet suspicious and send to detector;*
11. $P_k = (\lambda^N * e^{-\lambda}) / N!$

12. $H_{fd} = \lambda[1\text{-}\log_2 \lambda] + e^{-\lambda} \Sigma Pk$
13. ***If*** $H_{fd} > 0.5$ *then*
14. *Continue*
15. ***else***
16. *mark request packet suspicious and send to detector*
17. *reconstruction at victim V, in network N*
18. ***for each*** *coming packet p passing the reconstruction point mark recognition(length and fields)*
19. ***if*** *all fields in one entry are filled*
20. *output the source IP*
21. *delete the entry and drop the consumer request*
22. ***else if*** *same digest and segment number exist*
23. *create new entry*
24. *fill the address bits into entry and send to Double signature filter*
25. *Check for tag value*
26. ***If*** *tag value !=1*
27. *XML DDoS verify and drop the packet*
28. ***else***
29. *send the requested SOAP Message to cloud*

7 Experimental Result

7.1 Experimental Environment

Scalable simulation framework (SSFNet) is a collection of Java components used for modeling and simulation of IPs and networks. In experiment SSFNet Simulator has been used to simulate the whole process from embedding SOAP Header with signature to marking it. Darpa 1999 tcpdump (510 MB) dataset has been used as input traffic and analyzed it by using wireshark analyzer on a window7 OS, I3 Processor, 3 GB Memory and 300 GB hard disk.

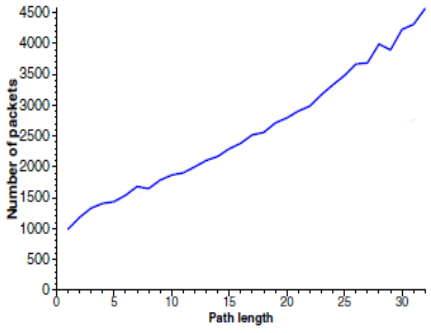

Fig. 5 Experimental results for number of packets needed to reconstruct paths of varying length

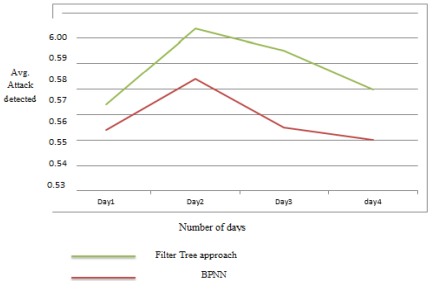

Fig. 6 Comparison between Filter Tree Approach and BCNN

7.2 Experimental Result

As in Fig. 6. simulation results, as red line showing in start BPNN takes time to make its system trained meanwhile most of the attacked packet enter in cloud so in start time less attack detected and as long as BPNN system gives knowledge to its system it detects more attacked packets but in Filter tree there is no need any previous knowledge so in start filter tree works well.

8 Conclusion

The denial-of-service attacks have become more targeted on cloud servicesand effected Client economically as eDDoS attack. This threat seems unlikely to fade away in absence of an active defense technique which pressures attacker's resources and raises the costs for delivering attack traffic. SOAP packet marking offer such a defense: An adversary cannot seize the victim's resources without committing its own resources first, which therefore limits its attack capability. Moreover Cloud Defender provides proactive defense approach. Intruder will get identify before get enter into the Cloud. Cloud defender will not check traceback for each request message, firstly it will identify the suspicious packet and will check only for those suspicious request packet. So this paper is filtering service request messages at different stages firstly matching the request client IP with previously stored suspicious IP in Trace-Back and then cloud defender is using for detecting the HTTP DDoS, Coercive parsing DDoS, XML DDoS at different stages. Cloud Defender is firstly identifying suspicious messages and then detecting attacks. This will reduce the computation cost and vurnability.

Proposed system works for HTTP interface further we can extend it to provide security for REST based APIs.

References

[1] Cloud Security Alliance (Online), `https://cloudsecurityalliance.org/topthreats` (viewed December 21, 2011)

[2] Europe Network and Information Security Agency (Online), `http://www.enisa.europa.eu/act/rm/files/deliverables/cloud-computing-risk-assessment` (viewed January 21, 2012)

[3] Microsoft Security Bulletin MS10 (Online), `http://www.microsoft.com/technet/security/bulletin/ms10-070.mspx` (updated October 26, 2011)

[4] Security of data (Online), `http://news.cnet.com/8301-138463-20052571-62` (viewed July 02, 2011)

[5] Security labs Blog (Online), `http://securitylabs.websense.com/content/Blogs/3402.asp` (viewed November 21, 2011)

[6] Nurmi, D., Wolski, R., Grzegorczyk, C., Obertellli, G., Soman, S., Youseff, L., Zagorodnov, D.: The Eucalyptus Open-source Cloud computing System, `http://www.eucalyptus.com/whitepapers`

[7] Bhuya, R., Ranjan, R., Calheiros, R.N.: Modeling and Siulation of Scalable Cloud Computing Environments and the CloudSim Toolkit: Challenges and Opportunities. In: Proceedings of the 7th High Performance Computing and Simulation Conference, Leipzig, Germany, June 21-24 (2009)

[8] Lin, C.-H., et al.: A Group Tracing and Filtering Tree for REST DDoS in Cloud Computing. International Journal of Degital Content Technology and its Applications 4(9) (December 2010)

[9] Tuncer, T., Tatar, Y.: Detection SYN Flooding Attacks Using Fuzzy Logic. In: International Conference on Information Security and Assurance, ISA 2008, April 24-26, pp. 321–325 (2008)

[10] Lu, L., et al.: A General Model of Probabilistic Packet Marking for IP Traceback. In: ASIACCS 2008, March 18-20. ACM, Tokyo (2008)

[11] Chonka, A., Xiang, Y., Zhou, W., Bonti, A.: Cloud security defense to protect cloud computing against HTTP -DoS and XML-DoS attacks. Journal of Network and Computer Applications 34, 1097–1107 (2011)

[12] Rahaman, M.A., Schaad, A., Rits, M.: Towards secure SOAP message exchange in a SOA. In: SWS 2006: Proceedings of the 3rd ACM Workshop on Secure Web Services, pp. 77–84. ACM Press (2006)