ISSN 2278-3091
Volume 9 No.2, March - April 2020
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse221922020.pdf
https://doi.org/10.30534/ijatcse/2020/221922020

A Novel Approach to Load Balancing and Cloud Computing Security using SSL in IaaS Environment

Bholanath Mukhopadhyay[1], Dr. Rajesh Bose[2], Dr. Sandip Roy[3]
[1]Brainware University, India, bnmukhopadhyay@gmail.com
[2]Brainware University, India, bose.raj00028@gmail.com
[3]Brainware University, India, sandiproy86@gmail.com

ABSTRACT

Cloud computing service providers (CCSP) are constantly at risk of suffering from performance loss. Cloud computing infrastructure allows users to rent computing resources at a fraction of the cost it would have otherwise taken to procure setup and maintain costly hardware and software systems. However, for a cloud computing service to stay relevant and enjoy the goodwill of its consumers, it needs to push the envelope of performance without sacrificing data security or vice versa. While Secure Socket Layer and related security algorithms have now come a long way since it was first introduced by Netscape very early on in the Internet age, there is a drawback that researchers have found associated with it. Performance degradation is a clear and present threat that cloud computing service providers are keen to avoid. In this paper, we present a proposed solution that is novel in its approach as we consider an existing commercial offering from F5, Inc., a renowned network equipment manufacturer, and incorporate its product – BIG-IP, into an experimental framework that promises to offer high availability, redundancy, load balancing and secure data channel simultaneously.

Key words: Secure Socket Layer (SSL), Local Traffic Manager (LTM), Access Policy Manager (APM), Infrastructure – as – a - Service (IaaS), Throughput.

1. INTRODUCTION

The ubiquity and widespread reliance on Internet have thrown up a host of challenges. Significant among those are the issues of security and performance. In the last decade of the 20th century, Netscape introduced Secure Socket Layer (SSL) to provide a secure tunnel for transmission of data in the Internet [1]. While SSL offers a bouquet of benefits for enabling secure transmissions over the Internet, the rates of transmissions are often slower [2] than when compared to unsecured transmissions [3]. Despite the advantages that SSL offers, several lacunae observed over the years forced researchers to investigate the efficacy of SSL. A new protocol derived from SSL was proposed and was eventually released in the public domain for use as Transport Layer Security (TLS) [4]. With the advent of Cloud Computing and rise in popularity of cloud computing services, the risks involved in data transmission have consequently grown higher. As a result, service providers and users have increasingly been reliant on SSL / TLS and VPN [5] services to secure data transmissions. Securing data in cloud is also considered to be a major challenge in ensuring that intermediate data is not lost or otherwise compromised in multi-tenant clouds [6].

To cater to increasing and humongous demands placed on servers that form the backbone of Internet services, the demands for Internet Data Centers (IDC) have risen proportionately. An Internet Data Center operates with a load balancing system to enable distribution and management of loads among its server nodes. Two decisive factors of using a load balancer is that it not only allows data center owners to enjoy benefits of long server hardware life but also brings about an improvement in the quality of service expected by users. A load balancer is a form of middleware that constantly coordinates and manages workloads placed on server nodes in distributed network architecture.

This, therefore, allows the whole network to not only self-regulate and heal in case of node failures and, thereby, improve fault tolerance but also ensure that network reliability standards are maintained. In case a server node experiences problems in functioning, its workload can be redirected to the closest available connected nodes with degrading network performance or compromising efficiency of the network [7].

To be efficient, a load balancer must ensure that a situation is not allowed to develop where several servers are busy and have a long list of jobs in queue while some other servers are allowed to remain idle. This makes task distribution by load balancer a challenging activity as it needs to select a server best suited for a task at hand while at the same time ensuring that a chosen server is not overloaded. A complex situation usually arises when it is time to ascertain whether a server is

overloaded. Usually, parameters such as CPU load, free memory and free file descriptors are considered while trying to determine whether a server can be assigned tasks.

Any IDC bereft of load balancer will experience an overall decrease in efficiency during peak load times. Power consumption may rise uncontrollably with a corresponding rate of slow performance. This is a situation that would neither be tenable for the operators of the IDC nor profitable. It is, therefore, in the best interests of all stake holders concerned that a proper load balancing system be put in place to handle multiple simultaneous requests with any perceptible lowering of performance and efficiency.

2. LITERATURE STUDY

2.1 Overview of Secure Socket Layer

A Secure Socket Layer (SSL) is a protocol that guarantees that data transmission between a client computer and server stays secure. At the time of establishing a secure communication between a client computer and a server, a public key encryption algorithm is used at both ends. The purpose of this is to ascertain whether the parameters of secret keys exchanged in the handshake are authentic. During the handshake protocol, authentication process involves three stages. The first being parameter negotiation, followed by mutual authentication and finally exchange of secret keys [8]. The client computers and servers use secret keys for encryption and decryption of data during transmissions [9].

The client PC begins initiation of connection through SSL handshake and transmission of a message that contains a list of algorithms for secret keys. These are known as cipher specs that are supported by client computers. At the other end, servers respond with a similar message and by choosing a cipher spec from its own list. The server also transmits a certificate containing its public key after sending the initiating message. That certificate is loaded with data about server's identity, public key and other parameters [8]. A Certificate Authority (CA) is responsible for generation of certificate and verification of its authenticity. To get a certificate, a server must go through secure channels to transmit to a CA its public key. Following generation of a certificate by CA, a message digest algorithm is used to produce a certificate fingerprint. The certificate that is produced by CA contains the CA's own identification details, details of the server's identification, server's public key and other relevant information. The message digest accepts a set of data stream and generates in turn an output of fixed length and deterministic in nature. The fingerprint is subsequently encrypted by CA using its private key to produce certificate signature. To be able to authenticate a server's certificate, a CA's public key needs to decipher signature and pre-calculated fingerprint. Separately, a client computer calculates the fingerprint of certificate independently. In case there is no match between the two fingerprints, it is assumed that unauthorized and unauthenticated alterations have been made to the certificate. To verify whether a server certificate received is authentic, a client PC checks to see whether the signature of the CA of the certificate matches with any one in the trusted list of CAs alongside their respective public keys. Soon after client and sever authentication process is complete and ready, both the client computer and the server can go ahead in determining secret-key by using public key algorithm. The data transfer stage of connection is initiated and entered with finished messages to finish handshake phase. At this stage, the client computer as well as the servers is required to indicate readiness by beginning use of secret keys [8].

While transferring data, client computers and servers convert outgoing messages into smaller tranches. These are then added with message authentication codes (MACs) which are fingerprints calculated and derived from data in messages. A MAC is essentially an encrypted message digest.

Either client computers or servers combine data fragments with corresponding record headers to form completed SSL packets by encrypting those with secret keys. Upon receiving such SSL packets either at the client computer end or at server-side, decryption is carried out, the MAC values are calculated, and the resultant values are matched with MAC values received.

A unique feature of SSL is resuming sessions. This was a result of SSL designers' efforts to address situations where computationally expensive and, consequently, time-intensive operations involving public key algorithms are required to be made with new connections between a client computer and a server within brief periods of time. Not only do such connections impose a heavy burden on servers, noticeable delays in response times are also evident at client computer end. The solution proposed rests on the ability of client computers and servers establishing unique session identities at the beginning of each new session. Subsequent connections with a set period or time frame could, thereafter, refer to corresponding session identities and use respective secret keys [8].

2.2 Load Balancing

Cloud computing has grown to be a dominant force in technology, economic, political, sports, news and entertainment landscapes. The ubiquity of cloud computing driven by the widespread proliferation of the Internet has made designers look closely at improving performance of cloud computing servers to ensure that two factors are balanced.

The first being consistently high user experience when accessing information on cloud platforms from any part of the globe. The second factor is to optimize resources that go into supporting cloud computing infrastructure.

Typically, a load balancer is placed between the external world and cloud computing data centers that serve user requests. A load balancer is configured with virtual servers that point to cluster of service points. In such scenarios, cloud computing infrastructure is setup with a return route pointing at the load balancer to route processed data traffic intended for client computers and users.
In general, a load balancing transaction has the following steps:

1. Users attempt to connect to services managed by load balancer.
2. A load balancer switches over to destination IP and matching port upon accepting a connection and ascertaining which server should an incoming request is to be routed to. The selection of the port is dependent on the service offered by the host server. The IP address of the originating request or client computer is not altered or affected in any way.
3. Servers in cloud data center accepts connection established by load balancer and responds via the same route.
4. On receiving return packets from host servers, the load balancer changes the source IP address and port to match those of the virtual servers in the packets that are to be transmitted to client computers.
5. The client servers receive their respective return packets which can only be traced back to virtual servers.

2.3 Load Balancing on the Network

The concept of load network-based load balancing forms the basis of application delivery controllers (ADCs). Since ADCs are can serve any application and operate independent of application servers, these boxes can achieve load balancing by relying on conventional network practices. Fundamentally, the ADCs expose only the virtual server address to outside world. Users connecting to ADCs find their respective connections forwarded to appropriately select physical servers are performing bi-directional network address translation (NAT) procedures.

Load balancers are designed to select appropriate servers for incoming connections based on health monitors and associated complexity of tasks. In case a physical server is not responding, load balancers would automatically re-route traffic away from the affected server until such time vital signs are received from it indicating that it has been restored to normal operations. Health monitors in network load balancers are not as feature packed as the ones that are developed by application developers. However, enhancements to hardware in network-based approach has significantly enhanced capabilities of load balancers to perform consistently and, thereby, provide unique virtual service entry points for application servers connected to load balancers.

Throughput in network-based load balancing approach is limited only by the hardware and networks connected. Although monitoring of health requires resources and performance compromises to be made up to a point, such does not expand exponentially as load balancers monitor health of clusters and not of each server. This has the effect of optimizing costs of networks and servers. High Availability (HA) can be made more robust by introducing hardware-based load balancers. On the other hand, load balancing hardware also must be deployed as HA pairs to ensure their own fault tolerance. Application neutral load balancing has been found to offer far more reliability and a stable platform solution that is worth investments made in network-based load balancing systems with HA [10].

2.4 Exploring Basic Load Balancing Terminology

In the context of load balancing technology, there are several terms and concepts which have physical manifestations. These may exist partially in certain load balancers while, in many other cases, all may exist at the same time. The terms of host, member, node and server require discussion from conceptual point of view in load balancing context. The node or server is a concept that defines the idea of a physical server that is setup to receive inbound traffic from load balancers. In the absence of a load balancer, the IP address of a physical server would normally be resolved to the name of the server. Throughout this paper, we shall refer to this concept as the host. The other concept that is integral to load balancing context is that of member. A member is also referred to as node by certain manufacturers; although, incorrectly in the opinion of one prominent manufacturer [11]. A member is that which receives inbound traffic and has a TCP port for the actual application that is configured to process requests.

2.5 Clusters, Farms and Pools: Distribution of Inbound Traffic

Organizations can distribute inbound traffic across a range of back-end processing points with the help of load balancing. As such, it becomes necessary to discuss the concept of back-end destinations and collection of processing points. These are collectively referred to as clusters and which shall be the term that shall be used throughout this paper. Clusters are also considered to be farms or pools offering consolidation of similar services accessible on several host machines.

2.6 Virtual Servers

The term "virtual server" was widely debated at one point of time. Today, the term for service definition, usually implies inclusion of application port along with IP address. Although, the term "virtual service" is more closely aligned to the conventional pairing of IP and Port, manufacturers and vendors prefer to use the term "virtual server" in network and load balancing architectures and designs.

2.7 Increasing Operational Efficiency in Networks using Local Traffic Manager

The purpose of Local Traffic Manager (LTM) is to provide high-end server efficiency and peak performance in networks through high performance and flexible application delivery architecture. Among the commercial offerings available in the market today, BIG-IP Local Traffic Manager by F5, Inc. has been a leading player in the field of cloud network infrastructure optimization to ensure high availability, robust security features and enhanced performance of applications in demanding business environments.

It is no secret that superior application performance boosts business operations and employee productivity leading to increased customer revenues. LTM enhances page load times thereby bringing about a positive change in user experience. This is made possible through real-time protocols and traffic management decisions that are influenced by actual server and application conditions, connectivity and TCP content onboard and offloading. BIG-IP LTM also offers enhancements by way of option add-on modules that enable enhanced performance.

In context of cloud computing, BIG-IP LTM has been equipped with F5 ScaleN technology that offers scaling capabilities that are responsive and sensitive to load demands. This enables cloud computing data centers to adapt quickly to application and performance needs. Considering that unavailability of applications during peak load hours can cause considerable damage in terms of goodwill and revenue to cloud computing service providers, high availability along with peak load distribution performance are considered key factors. The end goal of LTM is also to protect applications by providing redundancy and fine-grained bandwidth control for users. This leads to optimization of important applications. As a result of dynamically tracking of performance of individual servers in a group, LTM can ensure that network availability remains scalable and more manageable under all conditions. The following figure [Figure 1] shows a block diagram of LTM architecture.

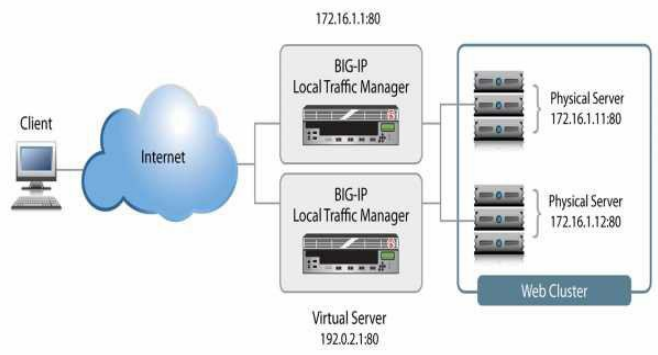

Figure 1: Block Diagram showing Local Traffic Manager deployment architecture

2.8 Role of Access Policy Manager (APM)

With the advent of globalization and rapid proliferation of the Internet, business and communities are no longer restricted exclusively to self-contained islands of information. Enterprises, organizations, medical institutions, educational institutions and even non-profit organizations now find that they are required to permit flow of data to outside world and even allow their users to access applications within and without the conventional boundaries of official and physical working environments. In many cases, however, users whether operating remotely or physically stationed within a given working environment, are allowed access to applications without authentication and authorization. This, therefore, necessitated that a central policy control be designed and developed to ensure proper and efficient management of a highly scalable, secure and dynamic work environment. F5, Inc. offers its BIG-IP Access Policy Manager (APM) to provide a highly flexible and secure high-performance solution that can extend a unified global access to any application hosted within a cloud computing framework. BIG-IP is bundled with context-aware access policies that are not only intuitive to understand and manage but are also well-designed to cater to almost all access policy management situations. This has resulted in BIG-IP APM turning out to be an invaluable toolkit in the arsenal of cloud computing service providers and helps to release valuable IT resources that aids in cost-effective and secure scaling (Figure 2).

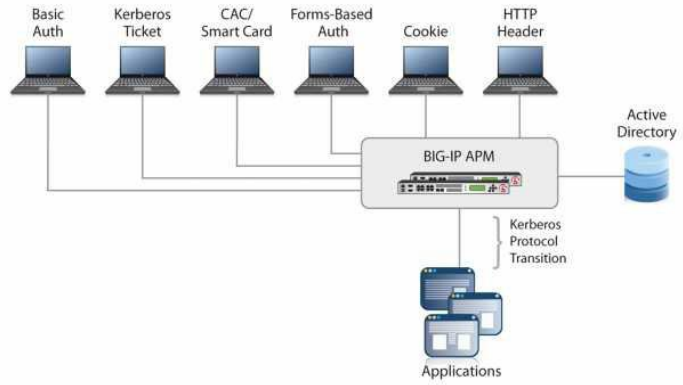

Figure 2: Block Diagram showing layout of BIG-IP's Access Policy Manager

2.9 Cloud Delivery Models

Cloud Computing has evolved to a stage where there can be several delivery models. However, the three main services model types are Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service (SaaS) [12], [13].

Infrastructure-as-a-Service

The fundamental basis of cloud computing lies in virtualization. Virtualization allows abstraction of required computing resources from a cluster of connected server and network hardware. Virtualization also allows hosting of multiple self-contained operating systems running their own set of applications independently on a same physical machine or server hardware. With Infrastructure-as-a-Service, users are free to choose the number of servers, computing resources required, file storage systems, operating software, and network infrastructure to set up their own software applications. This feature is extended to any user from a single service delivery point that enables seamless management and scalability of features and resources required.

In this form of cloud computing service delivery model, cloud computing service providers bundle in load balancers, virtual networks, firewalls, etc. either as part of standard package or as customized offerings that can be scaled on-demand. The very nature of elasticity in choosing hardware and computing resources required makes cloud computing an economically attractive and certainly a commercially sound and viable model.

Platform-as-a-Service

Ideally, this form of cloud computing service delivery model was designed to cater to software developers and their growing requirements to design, develop and deploy web applications without being burdened by the added weight of setting up and configuring the underlying infrastructure requirements. This makes for efficient use of time and management of resources as it this model obliges cloud computing service providers to take care of all associated computer hardware, operating system software, and networking paraphernalia including procurement, commissioning of devices and periodic renewal of annual maintenance contracts. For software developers who subscribe to this form of cloud computing service delivery model, the necessities of maintaining the underlying computing infrastructure required for software development are abstracted away at a price that is apportioned as cloud service rental fees that are paid usually on an hourly, monthly or annual basis.

Software-as-a-Service

Of all the service models available on the cloud computing platform, the most popular is Software-as-a-Service. The examples of such models could possibly number in their hundreds. The more common ones that are to be found in the virtual market space today are Google Docs, Microsoft 365, etc. With such models, users need only to concern themselves with functional features offered. In fact, this form of cloud computing service model was developed so that any user can use software that has been designed and offered across such model. End-users need not concern themselves as to how the software has been setup, nor do they need to know about the underlying hardware and computing infrastructure supporting such software operations.

While in the case of IaaS an end-user would require to be concerned about application and/or system software licensing deployment issues, in the case of SaaS or even in certain cases concerning PaaS, such are not a concern for consumers consuming PaaS or SaaS services. Considering that users can cut a wide swath through cost concerns in the absence of procurement and maintenance of underlying infrastructural requirements to support software application operations, it was not long before cloud computing became popular and consistently began growing ever more so with passage of time.

3. RELATED WORK

During our research, the extent and scope of existing corpus of work became evident. Sandhu et al. [14] had analysed the security and performance of several e-commerce applications. The authors describe their research findings after conducting experiments to measure the impact that SSL protocol has on request response times. Their research findings show that implementing SSL protocol resulted in mean client response times rising from 0.1 second to 6 seconds. Further, the costs of transmitting and receiving encrypted data were found to be, on an average, higher by 40% over raw data handling costs. Based on their research experiments, the authors were of the view that although requests with increased data transmissions were bound to be affected by secure connections enforced through SSL protocol, a comparison involving Apache Modules and Common Gateway Interface (CGI) revealed that the former did not cause server performance to improve significantly. Almeida et al. [15] in their research work revealed that there are several parameters involving SSL that affect performance of servers. These parameters range from throughput, cache sizes, utilization, cache miss rations, control dependencies, number of processors to many other vital metrics involving network loads, file access sizes, etc. The authors observe that the higher core frequency is directly proportionate to corresponding increase in SSL performance. Their research revealed that while a processor with high depth in pipeline can bring about a positive impact in SSL

transaction performance, any increase in width may not prove to be of significance in situations where bulk data encryption is carried out. Increasing L1 cache did, however, show that SSL performance was boosted. A complex logic combined with large BTB for branch handling is not likely to result in benefits in case of SSL transactions. Similarly, augmenting size of L2 caches up to a certain point is not likely to have significant impact on SSL transactional performance. The authors also noted that SSL handshakes and encryption / decryption operations of web pages with extensive content, exhibited good scaling properties corresponding to the number of processors in an SMP system. As a result of this inference, the authors state that 4-way or 8-way systems for such web applications are possible and can be explored.

In another [16], researchers observed and examined the maximum throughput obtainable for SSL using a range of combinations of cryptographic algorithms and a series of key lengths. The authors conclude that 1280-bit public key can be considered sufficient to ward off attacks from individual hackers. However, the authors also infer that a 1536-bit key would be needed to prevent against breaching attempts made by larger entities with malicious intent. The researchers also advise adopting 2048-bit public key encryption system to block attacks originating from hostile governments. The question of balancing performance and security has been at the forefront of many research studies. Application security through SSL has been the focus of research of one study [17] in which the authors explore the consequence of implementing security and the effects it has on performance. The results of their experiments reveal that SSL handshake is not without added costs and the resultant effects of additional security measures are dependent on the kind of architecture deployed for applications supporting business processes. Since analysing server performance to come up with better server designs is vital, researchers [18] have developed a System Monitor. Results of their experiments reveal that web servers spend almost 90% of the time, on an average, in processing HTTP requests when handling high numbers of client requests. A categorization approach to forward requests to appropriate web servers has been proposed in another research [19].

In this paper, we examine dispatch-based cluster or cluster-based web server systems as it is widely used. Incoming requests are processed by a centralized distributing entity in a cluster-based load balancing system. The centralized distributing entity, which is a part of web system, forwards requests to servers in cluster. The web system infrastructure is the only component that is under the control of content provider in a cluster-based web server framework. In a web server-based cluster architecture, a collection of web servers is connected to each other by high-speed networks and are locally distributed. Our proposed architecture shall have a single IP address to present before consumers. Users are neither required to know nor are they provided with the names and IP addresses of the physical machines that are part of cluster. Users shall access applications exposed to external world through cluster, by sending requests to a designated virtual IP address. F5, Inc.'s BIG-IP is deployed to act as a distribution or web switch. By examining the destination IP address – in this case, virtual IP address for requests arriving at cluster from outside world – routers will automatically route packets to distribution switch [20]-[22].

4. PROPOSED WORK

In this paper, we present our research to enable access to an application hosted on cloud, in a secure framework [23], [24]. We propose a solution combining F5, Inc.'s BIG-IP box at the gateway level and ahead of our IaaS deployment. For the purposes of our experiment, we deploy a virtual version of the product and utilize two of its core features to secure our applications. The first of these two features is the LTM that manages server load balancing as well as for extending SSL service. However, we propose two physical BIG-IP boxes for our experimental research. An SSL VPN that is built in to every web browser uses encryption and authentication technology to help establish a private virtual network across the Internet [25, 26]. For the purposes of our study, we propose to place BIG-IP between remote clients and application servers on cloud. A use attempting to access any hosted application on our proposed cloud computing service will be routed through an SSL VPN secure connection channel established by F5 box. Load balancing shall be conducted by F5 box, for selecting an appropriate server. Since, F5 box has built-in Access Policy Manager (APM) and Local Traffic Manager Module combined, seamless integration is possible. We also propose High Availability (HA) with two BIG-IP APM boxes in active-standby deployment mode. In case of a failure, an authenticated client session will be supported by BIG-IP features of LTM and APM (Figure 3). Requests immediately succeeding a failover shall be automatically forwarded to the active unit and session connections can be maintained without any visible form of interruption.

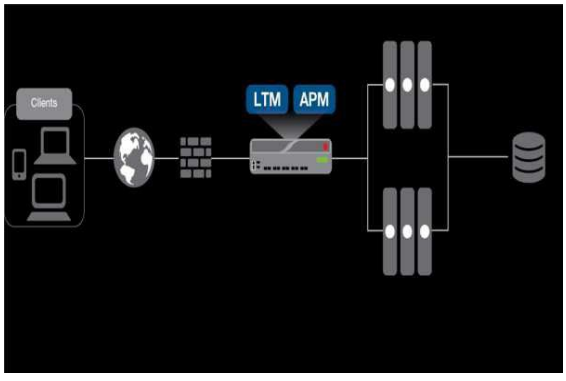

Figure 3: Diagram showing proposed layout of APM and LTM working as a composite unit or module

5. ANALYSIS OF SETUP

In our research experiments, we have considered two application servers. We also use virtual edition of F5, Inc.'s BIG-IP product to provide a secure environment for our application. We begin by installing virtual edition or OVF file provided by F5, Inc. in a VMware environment (Figure 4). In order to setup a secure access solution for our IaaS infrastructure, we use SSL VPN along with multiple secure access features such as network access, portal access including client and client-less features for our sample application hosted in cloud environment. For our experimental approach, we consider the following points:

1. Our proposed solution would be able to handle a minimum of 100 concurrent users.
2. Configuration changes are possible to grant various types of access depending on whether the access required is application, network or portal.
3. The URI is published on a portal to enable assorted SSO applications to access on either Linux or Windows operating system software.
4. Our proposed solution would be equipped with the capacity to handle load balancing duties with four application servers and 100 concurrent users.
5. Our proposed solution shall support HA with active-active configuration.

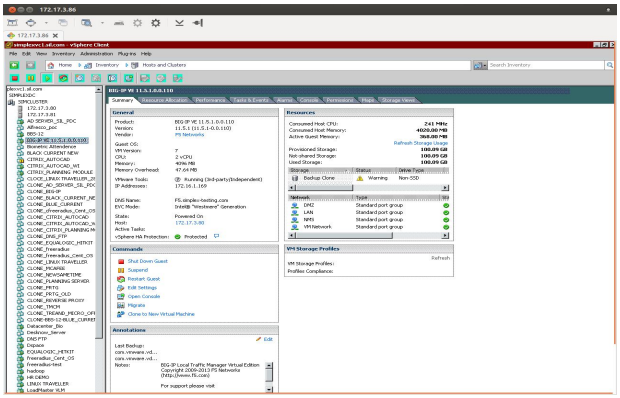

Figure 4: Installation of F5 .ovf file on virtual platform

For the purposes of our research experiments conducted in laboratory settings, we have used an evaluation license that allowed us the use of APM and LTM features. The screenshots (Figure 5 a – 5 d) that follow demonstrate how we configured our solution in IaaS deployment mode.

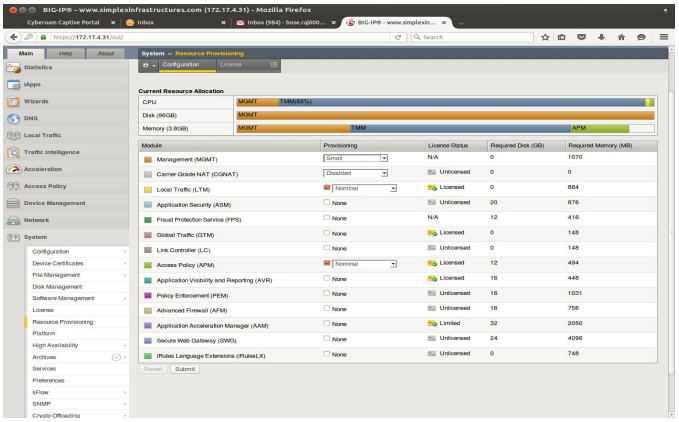

Figure 5 a: Step by step configuration of our solution in IaaS development mode

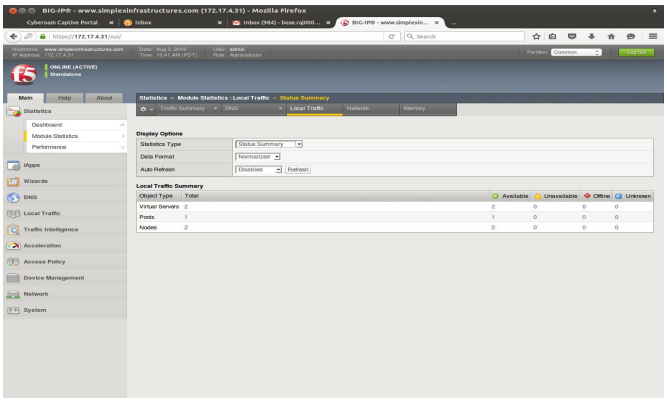

Figure 5 b: Step by step configuration of our solution in IaaS development mode

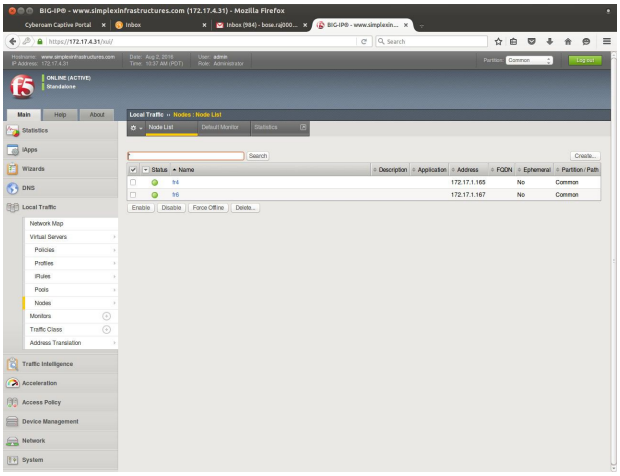

Figure 5 c: Step by step configuration of our solution in IaaS development mode

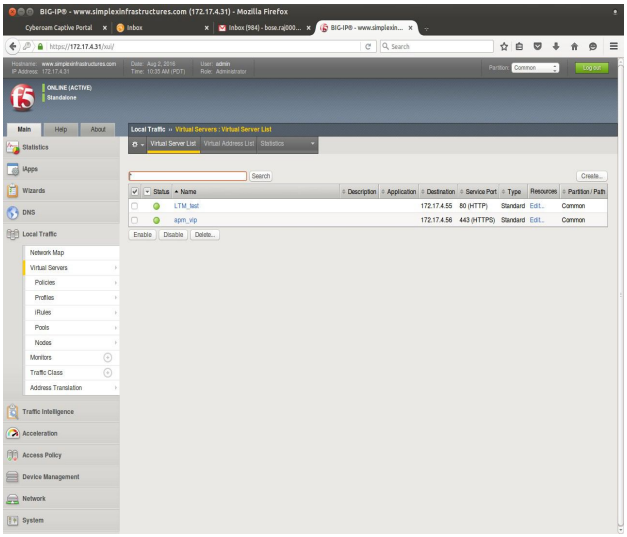

Figure 5 d: Step by step configuration of our solution in IaaS development mode

6. RESULT ANALYSIS

Our virtual server is always presented such that it remains accessible on client-side regardless of how it is used by servers to access gateway. The virtual server is positioned to extend service to clients and direct flow of data traffic from the direction of virtual server. This is illustrated in Figure 6. We can connect 100 users simultaneously to verify the throughput of our solution. Results of our experiment show that there is perceptible improvement in performance (Figure 7).

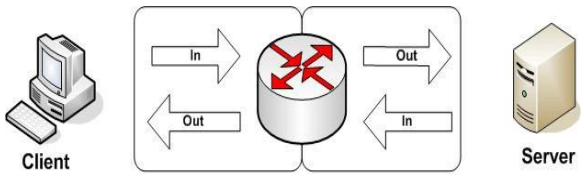

Figure 6: Inbound and out bound data procedure

Figure 7: Throughput Analysis Graph

7. CONCLUSION

Balancing security and performance have been a major obstacle for commercial ventures in cloud computing environments. F5, Inc.'s BIG-IP solution was chosen for our experiment as it is one of among the leading products in the field of APM and LTM. In our proposed solution, we had adopted a novel approach in which we designed access policies for authentication as well as for authorization. Alongside, we also incorporated the feature of an option check for endpoint security. In our setup, it is possible to create multiple profiles for a range of access methods each with its own unique access policy. For example, a web access authentication policy can be created for dynamic access control list (ACL) connections. Using our experimental approach, it is possible to quickly identify the user, location of user, prevailing network conditions at the time access was in progress, and condition of servers. Armed with these metrics and information, it becomes more advantageous to provide a critical shield for mission critical applications running in complex corporate environments.

REFERENCES

1. Beurdouche, B., Bhargavan, K., and Delignat-Lavaud, E. **Transport Layer Security (TLS)**, *2015 IEEE Symposium on Security and Privacy*, San Jose, CA, 2015, pp. 535-552.
2. Samad S. Kolahi, Yuqing Cao, and Hong Chen. **Impact of SSL security on bandwidth and delay in IEEE 802.11n WLAN using Windows 7**, *in Proc. 2016 10th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP)*, Prague, Czech Republic, 2016, pp. 1-4. https://doi.org/10.1109/CSNDSP.2016.7574043
3. Husák, M., Čermák, M., Jirsík, T., and Čeleda, P. **HTTPS traffic analysis and client identification using passive SSL/TLS fingerprinting**, EURASIP Journal on Information Security, no. 6, pp. 1-14, February 2016. https://doi.org/10.1186/s13635-016-0030-7
4. Sirohi, P., Agarwal, A., and Tyagi, S., **A comprehensive study on security attacks on SSL/TLS protocol**, *in Proc. 2016 2nd International Conference on Next Generation Computing Technologies (NGCT)*, Dehradun, India, 2016, pp. 893-898. https://doi.org/10.1109/NGCT.2016.7877537
5. Weil, T., **Risk Assessment Methods for Cloud Computing Platforms**, *in Proc. 2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC)*, Milwaukee, WI, USA, 2019, pp.545-547. https://doi.org/10.1109/COMPSAC.2019.00083
6. Wang, Y., Guo, Y., Guo, Z., Liu, W. and Yang, C., **Securing the Intermediate Data of Scientific Workflows in Clouds With ACISO**, *IEEE Access*, vol. *7*, pp.126603-126617, September 2019.

7. Loon, A.T.C. and Mahyuddin, M.N., **Network server load balancing using consensus-based control algorithm**, *in Proc. 2016 IEEE Industrial Electronics and Applications Conference (IEACon)*, Kota Kinabalu, pp. 291-296.
8. Kant, K., Iyer, R., and Mohapatra, P., **Architectural impact of secure socket layer on Internet servers: A retrospect**, *in 2012 IEEE 30th International Conference on Computer Design (ICCD)*, Montreal, QC, 2012, pp. 25-26. https://doi.org/10.1109/ICCD.2012.6378612
9. Lim, N., Majumdar, S., and Srivastava, V., **Security sieve: a technique for enhancing the performance of secure sockets layer-based distributed systems**. *International Journal of Parallel, Emergent and Distributed Systems*, vol. 31, no. 5, pp.481-503, August 2015. https://doi.org/10.1080/17445760.2015.1071367
10. Salchow, KJ (Ken)**, Load Balancing 101: The Evolution to Application Delivery Controllers.** *F5 Network, White Paper*, pp. 1-12, 2012.
11. Salchow, KJ (Ken), **Load Balancing 101: Nuts and Bolts**. *F5 Network, White Paper*, pp. 1-10, 2012.
12. Bose, R., Roy, S. and Sarddar, D., **User Satisfied Online IaaS Cloud Billing Architecture with the Help of Billboard Manager**. *International Journal of Grid Distribution Computing*, vol. 8, no. 2, pp.61-78, 2015. https://doi.org/10.14257/ijgdc.2015.8.2.07
13. Roy, S., Sarddar, D., **The Role of Cloud of Things in Smart Cities**. *International Journal of Computer Science and Information Security (IJCSIS)*, vol. 14, no. 11, pp. 683 – 698, 2016.
14. Sandhu, R., **Good enough security**, *in Proc.* IEEE *Conference on Internet Computing*, 2003, pp. 84-87.
15. Almeida, J.M., Almeida, V., and Yates, D.J., **Measuring the Behavior of a World-Wide Web Server**, *in Proc. International Conference on High Performance Networking*, Springer, Boston, MA, 1997, pp. 57-72.
16. **SSL Accelerator: A Technology Primer**, SonicWALL, pp. 1-15.
17. Das, M. L., Samdaria, N. **On the security of SSL/TLS-enabled applications**, Applied Computing and Informatics, vol. 10, no. 1-2, pp. 68-81, January 2014. https://doi.org/10.1016/j.aci.2014.02.001
18. Andrews, G.R., Dobkin, D.P., and Downey, P.J **Distributed allocation with pools of servers**. *in Proc. of the first ACM SIGACT-SIGOPS symposium on Principles of distributed computing*, 1982, pp. 73-83.
19. Colajanni, M. and Yu, P.S., **A performance study of robust load sharing strategies for distributed heterogeneous Web server systems**, *IEEE Transactions on Knowledge and Data Engineering*, vol. 14, no. 2, pp.398-414, March-April 2002. https://doi.org/10.1109/69.991724
20. Cardellini, V., Casalicchio, E., Colajanni, M., and Yu, P.S., **The state of the art in locally distributed Web-server systems**. *ACM Computing Surveys*, vol. 34, no. 2, pp.263-311, June 2002. https://doi.org/10.1145/508352.508355
21. Cardellini, V., Colajanni, M., and Yu, P.S., **Dynamic Load Balancing on Web-server Systems**. *IEEE Internet computing*, vol. 3, no. 3, pp.28-39, May-June 1999.
22. Brisco, T., **DNS Support for Load Balancing**. *Rutgers University*, pp. 1-7, April 1995.
23. Roy, S., Bose, R., and Sarddar, D**. Fuzzy based dynamic load balancing scheme for efficient edge server selection in Cloud-oriented content delivery network using Voronoi diagram**, in Proc. *2015 IEEE International Advance Computing Conference (IACC)*, Bangalore, 2015, pp. 828-833. https://doi.org/10.1109/IADCC.2015.7154822
24. Roy, S., Bose, R., and Sarddar, D**. A novel replica placement strategy using binary item-to-item collaborative filtering for efficient voronoi-based cloud-oriented content delivery network**, *in Proc. 2015 International Conference on Advances in Computer Engineering and Applications*, Ghaziabad, 2015, pp. 603-608. https://doi.org/10.1109/ICACEA.2015.7164762
25. Nagendra, K., Babu, A. S. **Enhancing Distributed Accountability by Using Proxy Re-encryption Scheme**, *International Journal of Advanced Trends in Computer Science and Engineering (IJATCSE)*, vol. 2, no. 5, pp. 09 – 14, 2013.
26. Ariffin, M. A. M., Rahman, K. Ab., Darus, M. Y., Awang, N., Kasiran, Z. **Data Leakage Detection in Cloud Computing Platform**, *International Journal of Advanced Trends in Computer Science and Engineering (IJATCSE)*, vol. 8, no. 1, pp. 400 – 408, 2019. https://doi.org/10.30534/ijatcse/2019/7081.32019