High Performance Clouds Computing with Encryption and Privacy

Jyoti
BRCM, Bahal

Abstract— High performance implements an event-driven, single-process model, which enables support for very high number of simultaneous connections at very high speeds. Multi-process or multi-threaded models can rarely cope with thousands of connections because of memory limits, system scheduler limits, and lock contention everywhere. Event-driven models do not have these problems because implementing all the tasks in user-space allows a finer resource and time management. The down side is that those programs generally don't scale well on multi-processor systems. That's the reason why they must be optimized to get the most work done from every CPU cycle. On security aspacts Encryption alone for Cloud privacy. Their classification hierarchy of Cloud Computing is not standard model and has few shortcomings, as we would discuss duly. We state the security and privacy issues from standard Cloud Computing definitions and discuss the challenges involved not just for FHE but also for many other techniques.

B. Outline

The rest of the paper is organized as follows. Section 2 contains a quick introduction to Cloud Computing delivery methods and deployment models. Section 3 Load Balancing and Proxy tools. Section 4 Security Chalanges 5 Proposed Solution 6 contains conclusions.

I. INTRODUCTION

Cloud computing came out of age. As it always happens, security is being an after thought. Before it is too late, its time for us to think through the tools available for us for secure cloud offerings.

To best of our knowledge, the proxy techniques for solving cloud computational privacy problems are microscopic, in the sense that, the protocols, schemes, mechanisms being devised solve a discrete subset of problems. We provide a panoramic view of privacy problems in various cloud delivery methods and current cryptographic technology landscape.

We emphasize the need for further generalization of different approaches and formalize the theory behind Computational Privacy for Cloud Computing.

A. Prior Work

With similar goals as stated in our work several companies worked on operating system capabilities for the good performances in the cloud infrastructure and internal communications in between instances. Those companies are providing professional and premium solution for cloud infrastructure E.g. Windows Azure and Ubuntu for cloud. A. Delivery Methods

The three delivery methods of Cloud Computing are

1) Software-As-a-Service (SaaS): In this method the user does not purchase software, but rather rents it for use on a subscription or pay-per-use model (an operational expense, known as OpEx). In some cases, the service is free for limited use. Example: Gmail, Google Drive, DropBox etc.

2) Platform-As-a-Service (PaaS): In this method, the service provider offers a development environment to application developers, who develop applications and offer those services through the provider's platform. Example: Google Gears , Microsoft Azure

3) Infrastructure-As-a-Service (IaaS): In this method, the service provider offers compute, storage and networking capabilities to the user. The user would be able to run any arbitrary software of his own including operating systems etc. Service provider at a remote place handles the physical infrastructure and virtual abstractions are given to the user. Example: Amazon Web Services, Google's Compute Engine.

B. Deployment Models

The deployment model for the discussion throughout this paper would be Public Clouds (and also Hybrid Clouds). In Public Cloud deployments Cloud platform cannot be relied upon as the cloud infrastructure is run at service provider premises and open for public use. In Hybrid Clouds too part of the cloud infrastructure is run at service provider. Whereas in Private Cloud deployments the platform can be trusted since its completely within users premises.

II. CLOUD COMPUTING

Cloud computing has been standardized now. The principles defining the essential characteristics, delivery methods and deployment models are now well laid and widely accepted [1].

III. LOAD BALANCING AND PROXY TOOLS.

Load Balancing: -

Load balancers are integral part of the Linux and Unix Operating System. Load balancing is a computer networking method for distributing workloads across multiple computers or a computer cluster, network links, central processing units, disk drives, or other resources. Successful load balancing optimizes resource use, maximizes throughput, minimizes response time, and avoids overload. Using multiple components with load balancing instead of a single component may increase reliability through redundancy. Load balancing is usually provided by dedicated software or hardware, such as a multilayer switch or a Domain Name System server Process.

Proxy Server: -

A proxy server is a server that acts as an intermediary between a workstation user and the Internet so that the enterprise can ensure security, administrative control, and caching service. A proxy server is associated with or part of a gateway server that separates the enterprise network from the outside network and a firewall server that protects the enterprise network from outside intrusion.

Hyper Proxy: -

Hyper Proxy is a concept solution offering high availability, load balancing, and proxy for TCP and HTTP-based applications. It is particularly suited for web sites crawling under very high loads while needing persistence or Layer7 processing. Supporting tens of thousands of connections is clearly realistic with today's hardware. Its mode of operation makes its integration into existing architectures very easy and riskless, while still offering the possibility not to expose fragile web servers to the Net, such as below: -

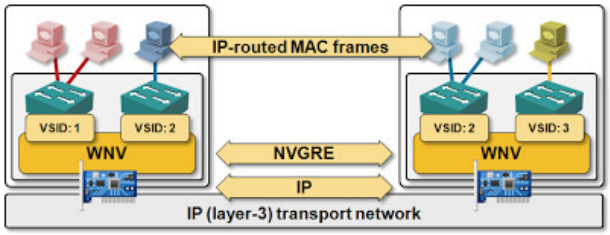

Fig : [1]

IV. SECURITY CHALLENGES

Many variants of computational privacy problems are being formalized cryptographically and solutions are being proposed. Success has been achieved in various degrees in many of them. We give a whirlwind tour of those techniques here. And discuss the challenges associated with each of these for adapting to Cloud Computing.

A. Instance hiding (IH)

If a user wants to outsource the computation of a function for a particular input x (instance). She transforms the input x to an encrypted input y (thus hides it) in such a way that the server cannot infer x from y and sends to the server. The server computes the function on y and returns the result. The user then transforms the result f(y) back to the value of f(x). These techniques are called Instance Hiding techniques [18] as they hide the actual inputs from the server. The functions that can be evaluated this way are called encryptable functions.

Few protocols were also proposed to achieve operational privacy [19] using these techniques.

Prima Facie these techniques look they can be adapted for Cloud setting. But it has been proved that not all functions are encryptable, this means not many functions can be evaluated when the real input instances are hidden from the server.

If there aren't many encryptable functions then the results look contradicting with recent breakthroughs of FHE schemes. FHE schemes aim to perform generically all functions by computing fundamental operations like add, multi on transformed inputs. Of course there is no formal analysis done on connections between both of them.

B. Specialized Operations

a) Proxy re- encryption: techniques allows to translate a cipher text encrypted under one key to cipher text encrypted under another key without every decrypting it, provided some additional information [2], [3]. These techniques are used in distributed secure storage.

b) Searchable encryption: techniques allow performing search over encrypted data [4], [5]. These techniques have been improved and implemented in MIT's crypt-db project [6].

c) SQL-Aware encryption: is a strategy rather than a technique in itself. Its based on the fact that all SQL Queries are made up of well defined primitive operations like add, equality, order check etc. So a collection of encryption schemes that allow these operations have been engineered into an RDBMS application. This made possible to execute SQL-like queries on encrypted databases [6].

These specialized techniques cater to small subset of functionality that can achieve. Finding connections and realizations of these specific techniques with much more generic techniques like FHE or FE might give us insights into possible efficient solutions.

V. PROPOSED SOLUTION

With single process architecture, Hyper Proxy lowers the cost of context-switches and memory usage while not locking up resources with long blocking operations.

Managing this tool is really very easy for any scale up infrastructure. Which have Unix and Linux base system. If you have a Unix based system, you can test-run Hyper Proxy yourself on your own machine: it may either be packaged for your favorite system you can download the source code and compile it yourself. In the case of compilation, the result consists of single binary and a possible configuration file is very lean and minimal.

The configuration file can be as simple as:

```
  global
      daemon
      GITHSR001 256
defaults
      mode http
      timeout connect 5000ms
      timeout client 50000ms
      timeout server 50000ms
  frontend http-in
      bind *:80
      default_backend servers
  backend servers
      server server1 127.0.0.1:8000 GITHSR001 32
```

This configures Hyper Proxy to listen on port 80 for HTTP requests and proxy those to server1 on the port 8000 of the same machine.

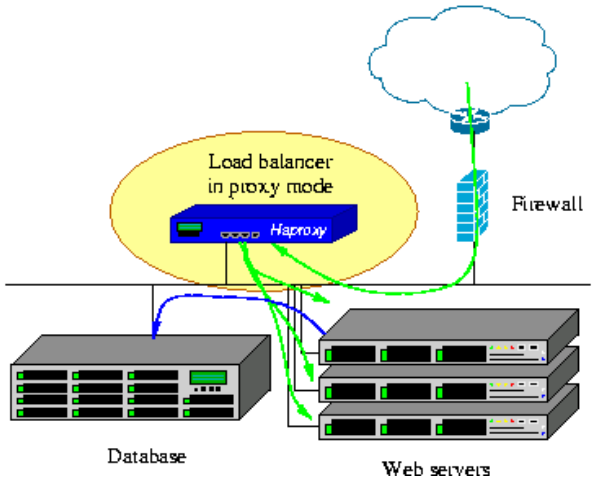

Fig : [2]

Hyper Proxy & OpenShift: When a request comes, it is translated by OpenShift's load-balancing layer from the domain name (Host: HTTP header) into an internal IP address and is proxied there.

Fig : [2] Shows the proposed solution using proxy and load balancer as one instance and all other server instances will be connected with proxy directly but data and N.A.S. will have separate dedicated connection with the process instance. Firewall Security will be at the top of Proxy for centralization security management system at filtering incoming connection and request handling.

Then the HTTP request hits the main gear on port 8080 where Hyper Proxy is listening. It is configured in a way that makes it aware of all the gears that are provisioned for that particular application. Hyper Proxy's task is to load-balance the request among all those gears, and that is also the use-case it is designed for.

As your application scales up and down, the Hyper Proxy is automatically configured by OpenShift to be aware of all actively deployed gears.

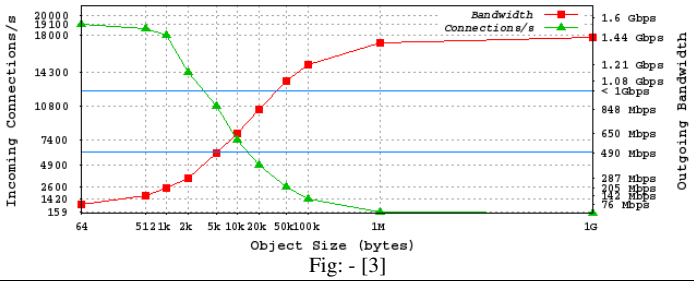

Fig: - [3]

SR.	INCOMING CONNECTION	OBJECT SIZE IN KB	OUTGOING BANDWIDTH @ N.C. GBPS
1	19000+	0.8	1.53
2	18000+	0.5	1.50
3	18000	1	1.44
4	14300	2	1.93
5	10800	5	.820
6	7500	10	.603
7.	4900	20	.430
8.	2600	50	.205
9.	1420	100	.142
10	1024	159	.076

Table :- 1

[Fig: - [3]] contains the result from the testing of proposed solution with below mentioned configuration. [Table:- [1]] contains the summary of results which helps in concluding the research.

As per the test result shown above the increasing the object packet size will resultant in to the high performance and lower bandwidth uses. Due to limitation of testing tools we can max put the load of receiving packets 1.7 GBPS test result is with one server and 6 instances of virtual servers in it running with hyper proxy. Same solution implemented with the multiple server transmitting the packet size 1M or above will lead to high performance due to lower switching at proxy for packet handling.

Thus the user does not need to be know that scaling is happening and also the application developer can sleep peacefully know that OpenShift is handling all the configuration automatically.

VI. CONCLUSION

Hyper Proxy tools HAProxy and Cloud Computing functionality tools like Redhat's Concept Project RHClouds can be clubbed together for achieving High Performance Public and Private Cloud Infrastructure. For Privacy we need to implement the encryption at DataBase Layer. At the Gars using OpenShift and HAproxy on Ubuntu we are able to achieve 7GBPS system network transfer using IBM X350 Rack Servers with 2-Physical XEON Macro Processors and RedHat Enterprises Operating System. With OpenShift automates the configuration of HAProxy and thus relieves the developer (or deploying sys admin) of the need of learn another piece of infrastructure.

ACKNOWLEDGMENT

I would like to thank GARS Infotech for supporting this work. I would like to thank my research advisor Dr Amit Kumar at BRCM-Bahal for all the insightful discussions and brainstorming. I would like to thank Mr.Sudesh Kumar for all the opportunities he created for me to learn cryptography and Cloud security in detail.

REFERENCES

[1] P. Mell and T. Grance, "The nist definition of cloud computing, special publication 800-145," US Department of Commerce, Gaithersburg, MD, 2011.

[2] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," in Advances in CryptologyEUROCRYPT'98. Springer, 1998, pp. 127–144.

[3] R. Canetti and S. Hohenberger, "Chosen-ciphertext secure proxy reencryption," in Proceedings of the 14th ACM conference on Computer and communications security. ACM, 2007, pp. 185–194.

[4] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in Security and Privacy, 2000. S&P 2000. Proceedings. 2000 IEEE Symposium on. IEEE, 2000, pp. 44–55.

[5] M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Paillier, and H. Shi, "Searchable encryption revisited: Consistency properties, relation to anonymousibe, and extensions," Journal of Cryptology, vol. 21, no. 3, pp. 350–391, 2008.

[6] R. A. Popa, C. Redfield, N. Zeldovich, and H. Balakrishnan, "Cryptdb: protecting confidentiality with encrypted query processing," in Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles. ACM, 2011, pp. 85–10