I J C T A, 9(3), 2016, pp. 1641-1646

Designing a Wrapper for Secure Execution of Console Applications in a Distributed Environment

Alok Rajiv* and Madiajagan**

Abstract: Deployment of resources into scalable services is the primary step involved in product deployment. This often requires the involvement of an expert team because exposing an existing application as a service for interoperability, or public cloud deployment for end users, involves complications that are quite different from the application development domain itself. However, this implementation is quite common in its specifications and design which brings in the possibility of creating a generic version of such a wrapper that will be able to wrap around existing applications to magically allow them to scale and deploy. This research paper attempts to find a feasible solution to address the complexities in the design of a secure wrapper for executing applications in a distributed environment.

Keywords: Distributed Replication, DMZ, Generic Plug-ability, Health Monitoring, Scalability, Resilient, Wrapper

1. INTRODUCTION

The major technology players are investing heavily in BIG DATA, DATA MINING, IOT and CLOUD which are the four big pillars of the Information Technology Era as they are challenging each other to lead the new technology revolution. However, something that has been very importantly mentioned by many great minds is that expertise is the specific and very often we find people specializing in one of the four of these and very often much narrower into specific streams inside each of these platforms. However, this platform provides the researchers to come out with a design technique for integrating one of these pillars directly into others and thus avoiding competition and enhance collaborations with each other. The cloud technology is the oldest of four pillars and is still rapidly growing in exponentials in terms of infrastructure and investment.

2. IMPORTANCE OF APPRAOCH

This paper proposes a design for the wrapping up of Niche applications around an existing wrapper and then using the proposed wrapper as the engine or server for all communication and inter-operational capabilities. This approach is specifically useful in the IT sector because of the amazing level of specialization being put into this industry. While they are very independent domains by themselves, in most use cases of production environments they require to collaborate and integrate for effective delivery. The proposed design has the following characteristics: deliverability, security, scalability, abstracted encapsulation and generic plug-ability.

2.1. Deliverability

This feature allows for the delivery of applications and deployment which is one of the most important concerns for developers. In scenarios like data-mining algorithm implementations it is very often that deliverability is not a concern until the algorithm has matured enough. The complications of deliverability

* Sr. Undergraduate Student, BITS Pilani, Dubai Campus, Dubai, *Email: f2013119@dubai.bits-pilani.ac.in*

** Asst. Professor & Prof. in Charge ICT, BITS Pilani, Dubai Campus, Dubai, *Email: madiajagan@dubai.bits-pilani.ac.in*

arise when converting an application to an interoperable-service, are often not pertaining to the same domain as that of the researcher and requires separate workforce to tackle them. However, deployment of resources as services is not new and has existed for more than a decade making unnecessary repetition of work a waste of time and resources.

2.2. Security

Exposing anything to unpredictable environments results in security complexities that is very critical to the system. The application layer security is a parameter where experts have to keep in mind during development but du-diligence has to be also given to security of components which are parts of the cloud deployment infrastructure. A standardized platform for wrapping will enable a standardized solution to such problems and thereby during inclusion have security implementation brought along with it.

2.3. Scalability

Distributed Setup is an important implementation technique used for scaling applications today for performance boost [1]. Multi-Node systems with Distributed Scaling techniques for massive scalability enables for organizing and distributing work and increasing this often directly enable for increasing performance of the system together.

2.4. Abstracted Encapsulation

The wrapper has the ability to abstract away the secure feature implementations and give an encapsulated view or control for the end user, to safe guard mishaps in critical components. Distributed execution concerns including the scalability and security aspects included perhaps prove to be very useful to researchers and programmers who might be focusing on other concerns in their domains which are mostly different from this. The abstract approach is specifically useful for systems also which simply do not out of the box understand distributed architecture and cannot optimize itself in such environments to gain performance boosts and optimizations.

2.5. Generic Plug-ability

The Generic mechanism to interface with any type of application by maintaining a standard communication protocol at required end points for Input and Output is an integral part of the wrapper. Developing a common platform for different programming languages, constraints and styles is itself a research area for creation of pluggable applications. Though there are many answers by constraining our interfacing to console applications we have the a acceptable method-the console either through bash or command-prompt. This end-point will be an easy-approach interface by applications in a generic sense while this approach has its own strengths and weaknesses.

3. WRAPPER DESIGN ARCHITECTURE

The schematic block-diagram of the proposed application design architecture is shown in Figure 1. The applications generic driver/software sitting on top of the distributed system acting as the head-node to execute and control applications running in the distributed setup. The data nodes and the computation nodes can be replicated in distributed setup that enhances scaling feature of the application. The architecture pattern to design the wrapper adopts a

3.1. Intercomm-Layer

This layer is more of a de-militarized zone for external access or end-user access to the underlying applications. Any request from outside services, or requests required by the internal systems goes through

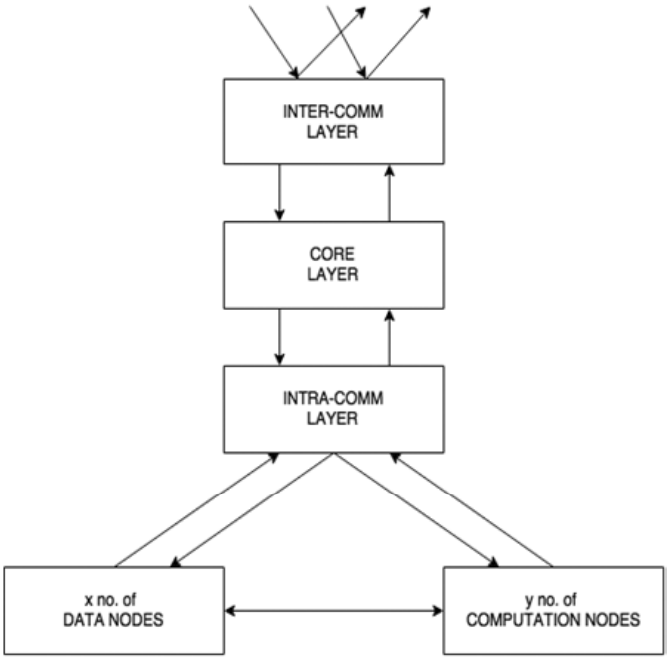

Figure 1: Schematic Block-Diagram of the wrapping System

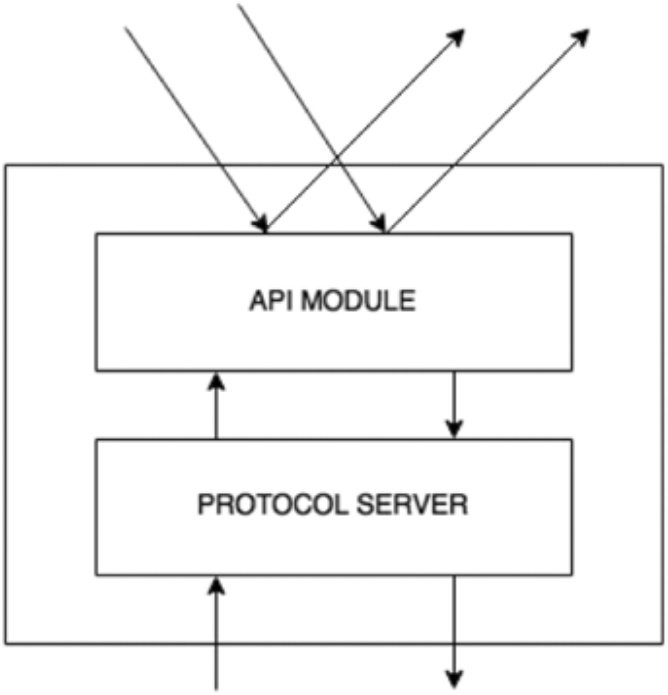

Figure 2: Inter-comm Cross-Section

this layer making it a proxy in a demilitarized zone. The region below this layer can be protected using an IDS or IPS Systems to allow only the authenticated layer protocols to seep through. Thus, in case an attacker is able to breach in, he cannot get access to any of the critical systems below but instead be restricted to the demilitarized zone which mostly stores externally visible data only.

There has to be Service End-Point Contact for other applications or Users in the DMZ region because it is the most externally exposed region of this system. The API can be written in any language but should understand and serialize requests or data into a standard form for the system. APIs are quite a common feature in most cloud integrated applications with the language medium almost always being HTML, JSON or XML though other languages such as YQL, TXT are also used. In-short, this layer is basically a translator to external entities in a standardized protocol or medium. A server that takes care of what protocol communication the system likes to operate with should be attached. It could be multiple protocol servers for communicating in different protocols at the same time or a single dedicated daemon for a single protocol alone.

3.2. Core-Layer

A Core Layer being the brain of the entire operations contains most of the crucial function implementations. These include the supervisor which is responsible for most decisions that happen inside the platform. This layer needs to take care of many intensive operations being executed inside the wrapper system, specifically in respect to monitoring and control. All the connections to this region critical to the working of the entire system because administration of the platform will be handled from this layer. The operations could be automatic or manual but this system-layer has the highest execution privilege inside the platform because all decisions are taken here and instructions to other modules to comply based on the decisions made originate from here. Monitoring of health, and logging in general has to be done here and is crucial for ensuring performance correctness [4]. It has to be done with special importance for real-time statistics and logging for active or passive diagnostics. Scheduling operations for load balancing and optimizations also requires analysis and intelligent decision making based on data within this diagnostic logs. The scheduling of instances to finish a request from an outside user or service is actually processed here. The distributed

computation and storage nodes will be available as layer below with the supervision to be performed from here though the core layer takes care of what is executed where for optimal performance. Generally speaking, most monitoring operations that will be done in the core layer are also autonomously in execution in Compute Nodes and Storage Nodes as well. This allows for a centralized command for all crucial operations while less priority operations are run in a rather more decentralized manner.

3.3. Intra-comm Layer

The inter-comm layer takes care of all communications to distributed devices, nodes and services which are available within the platform. This layer will be associated with a physical system because it takes care of internal routing as well as being the communication gateway. It is possible to do this using a software router but flexibility to extend this feature in both ways exists depending on the nature of Datacenter or set-up which might be required to run the platform.

The platform intra-communication latency issues are to be taken care by the intra-comm module of the wrapping system. It is common for highly efficient and distributed systems or platforms to face latency issues in communications. Usually Hard Disk drives are a cause of these bottlenecks because they cannot cop up with the high requirements of a highly operational distributed setup [5]. Though SSDs have solved this problem to some extend, network I/O is still a big bottleneck for distributed-memory based setups [6]. The solutions like Gigabit I/O could be the answer to this problem which otherwise results in Computing or memory wastage when arguably these are considered to be more expensive or valuable resources.

3.4. Data Nodes

The data is of crucial importance of all kinds of computational models, though distributed systems has to taken care of this with special attention. Occurrences like node failures which require as automated recovery effort from the system implicitly as well as issues like data-corruption which is often considered common occurrence in distributed systems [7] have to be kept in mind. Therefore, what is required is resilient and intelligent systems from base up which can handle such critical and predictable issues with minimal or no human intervention.

Based on the popular software reusable approach, well established systems such as the Hadoop File System will be a very good idea for file-system implementations. The Hadoop File System (HDFS) has multi-level file system API support which can be used for data storage ensuring Reliability and Security integrations because they are already available out-of-the-box from the File-System protocol itself. Other emerging platforms such as File Service Project of Open Stack, Manilla are one of many other intelligent components that can be brought into this module as well.

3.5. Computation Nodes

The computation nodes are the workers in the platform and represent all of the computation and primary memory resources available inside the system. The computation nodes can behave as an asymmetric cluster or symmetric cluster based on the existing infrastructure. The computation nodes have an instance containing many important functionalities such as Status Reporting, Environment Setup, Disaster Recovery, Situation Reporting to the centralized components in the Core Layer.

Since, Computation Nodes are the once which run the applications, they have to be very resilient which can be achieved using state saving or other virtualization procedures as well. The other approach is to combine the physical resources together into a cloud and create virtual nodes depending on the required environment or setup. This approach will also be advantageous for node redundancy and state replication for cloning, scaling or live migration scenarios.

4. SCALABILITY SPECIFICATIONS

Scalability can be addressed by distributing the compute and storage over multiple nodes which could be physical nodes or virtualized systems. The platform runs instances of the application in different nodes to share and balance workload for optimal performance. The replication and management of instances in a distributed environment is something that is abstracted and carried out by the wrapper. However, implementation of such a system requires serious thinking for complications that arise in regard to performance, inter-operability, cost and ease of setting up and maintenance. Monitoring of operations and control decisions taking place inside the platform are also serious issues that needs to be planned and taken care off[8].

5. SECURITY SPECIFICATIONS

Security plays crucial role in any system where multiple services or users are sharing same or connected resources. Being an abstracted wrapping platform, the security of the wrapper itself and its connections are of primary importance because all external connections are routed through it. The 3-tier architecture is the base model of design with security provisioning using a Demilitarized Zone (DMZ) [9]. The first tier of the architecture represents the demilitarized zone and is available to the external services. All other services are connected behind the DMZ which requires separate authentication and firewall protections. The IDS and IPS of the Firewalls would be able to protect the tiers and remove critical regions from direct contact to external resources. During an attempted security breach, DMZ can get compromised but the below tiers will be protected from the breach. Autonomous defense systems can defend a DMZ from attacks like DDoS [10] and try to recover while services can be redirected through another secure channel or zone.

6. CONCLUSION

An architecture for the wrapping system has been discussed in this paper and its modules have been identified and well defined with their functionalities. The present work can be further extended with the detailed architectural design to showcase the operational work flows and implementation specifics and develop a system which is secured and protected from all kinds of attacks like injections and DDoS. The proposed design can be very much helpful in practical applications such as distributed deployment implementations in cloud services for high availability.

Acknowledgment

The authors would like to thank Professor Dr. Santosh Kumar whose valuable comments and arguments helped shape the design and logic for this technology.

References

[1] André B. Bondi “Characteristics of scalability and their impact on performance” WOSP ’00 Proceedings of the 2nd international workshop on Software and performance ISBN:1-58113-195-X, Pages 195-203.

[2] Ashish Seth, Himanshu Agarwal, Ashim Raj Singla “ACM SIGSOFT Software Engineering Notes archive Volume 36 Issue 5, September 2011” Pages 1-7.

[3] Randy W. Connolly “Complecto mutatio: teaching software design best practices using multi-platform development” ITiCSE ’08 Proceedings of the 13th annual conference on Innovation and technology in computer science education ISBN: 978-1-60558-078-4 Pages 345-345.

[4] Shimin Chen, Phillip B. Gibbons, Michael Kozuch, Todd C. Mowry, Carnegie Mellon University “Log-based architectures: using multicore to help software behave correctly” ACM SIGOPS Operating Systems Review archive Volume 45 Issue 1, January 2011 Pages 84-91.

[5] Yoji Yamato “Use case study of HDD-SSD hybrid storage, distributed storage and HDD storage on OpenStack” IDEAS ’15 Proceedings of the 19th International Database Engineering & Applications Symposium ISBN: 978-1-4503-3414-3 Pages 228-229.

[6] Hemy, M., Steenkiste, P. “Gigabit I/O for Distributed-Memory Machines: Architecture and Applications” Supercomputing, 1995. Proceedings of the IEEE/ACM SC95 Conference ISBN: 0-89791-816-9 Page(s):58.

[7] Peipei Wang, Dean, D. J., Xiaohui Gu “Understanding Real World Data Corruptions in Cloud Systems” Cloud Engineering (IC2E), 2015 IEEE International Conference INSPEC AN: 15090130 Page(s):116–125.

[8] Mauro Andreolini, Michele Colajanni, Riccardo Lancellotti “Assessing the overhead and scalability of system monitors for large data centers” CloudCP ‘11 Proceedings of the First International Workshop on Cloud Computing Platforms ISBN: 978-1-4503-0727-7 Article No. 3.

[9] Mick Bauer “Paranoid Penguin: Designing and Using DMZ Networks to Protect Internet Servers” Linux Journal Volume 2001 Issue 83es, March 2001 Article No. 16.

[10] Sardana, Anjali; Joshi, R. C. “Autonomous dynamic honeypot routing mechanism for mitigating DDoS attacks in DMZ” Networks, 2008. ICON 2008. 16th IEEE International Conference on , Issue Date: 12-14 Dec. 2008.