ISSN:0975-9646

Scalability & Security Architecture for Internet2 Applications

T.Mahesh, Prof. P.Pradeep, R.SatyaTeja

Department CSE, Vivekananda Institute of Technology and Science, Karimnagar, AP, India

***Abstract*-The last two decades have seen a tremendous change in the field of Internet services. The main reason for this sea change is the impact of Information and Communication Technology tools and services which have influenced the Internet services. For the last decade, each new technological advance has predicted the demise of brick and indeed we have seen many giants fall. Through various technologies like cloud computing, Web 3.0, grid computing, and applications will have to adopt novel software engineering techniques for development and deployment. Scalability, Interoperability, Availability (tolerance to failure), Reliability, Security (confidentiality, integrity, authentication, authorization), and Anonymity are some of the important quality and usability features that will determine the success of new Internet applications. Internet2 actively engages our community in the development of important new technologies including middleware, security, network research and performance measurement capabilities which are critical to the forward progress Internet applications. This paper captures the most recent and innovative architecture and mechanism that will enable to successfully build the Internet2.**

Keywords: Scalability and Security aware Software Development Life Cycle, Cloud Computing, middleware, web 3.0, Internet2

I. INTRODUCTION

With ever-increasing demands on capacity, quality of service, speed, and reliability, current internet systems are under strain and under review. Web and Smart phones have changed the rules of the innovation game – innovation in technology has moved from technology companies to the consumer space. Innovation in content delivery, multimedia services, social networking services, Web 2.0, Web 3.0, virtual reality applications, and context-based services are creating both opportunities and challenges for business and academic researchers. Internet2 applications and services will need to answer all these questions and many more.. Moreover, the applications will be available to novice inexperienced users and expert hackers as well. The success of these Internet2 applications and services will depend on how are they created, deployed, and maintained. For a long time, software engineering dealt with only functional requirement. Different techniques have been proposed in last four decades to define software development as a mainstream engineering practice. *Unified modeling language* (UML) [1] is the most significant contribution in this attempt. UML provides various tools to elicit functional requirements and design. These tools help reduce error at the early stage of the application development. However, UML does not include nonfunctional requirements like security, scalability, reliability, interoperability, and availability (tolerance to failure), which are difficult to quantify and hard to define in measurable terms; therefore, they remained nonfunctional requirements and left to the programmer to meet these complex challenges based on their knowledge and judgment. Also, quite often these nonfunctional requirements were addressed outside of the application domain. When the application fails to scale, additional hardware are added to address the performance issue. Security was also addressed outside of the application through perimeter security. For Internet2 applications these nonfunctional requirements must be analyzed at the early stage of the *software development life cycle* (SDLC) and included inside the application and treated as functional requirements. Some of the recent works [2, 3, 4, 5, 6] recommend inclusion of nonfunctional requirements into mainstream software engineering.

Cloud computing is a paradigm of service deployment that is gaining lot of interest and momentum. Cloud Computing helps users to rent computing resources during deployment of the service against owning it in pay as you go model. Cloud helps converting capital expenditure into operational expenditure; also, cloud computing can be used to channelize unused computing resource in the enterprise reducing the power requirement in the data-center. An application may run on any platform or any virtualized infrastructure through infrastructure as a service (IaaS), platform as a service (PaaS) and software as a service (SaaS) [7]. Companies like Google, Amazon, Yahoo, Facebook etcetera are using cloud in some form or the other – they are also offering cloud infrastructure for public use. Many users have started using cloud – be it a business application or a portal. Cloud-ready services will be exposed to adversaries in un-trusted networks and will exchange data over these networks. Also, they will migrate from platform to platform at different point in time based on the economics of the resource. The question that need to be answered in Internet2 – how to ensure that the application is secure and scales up or down as and when necessary; how can this be made infrastructure agnostic and platform agnostic and provide the same level of security, scalability, and availability as in a private environment. All these need a paradigm shift; scalability and security can no longer remain outside of the SDLC. The standard UML way of designing an application will not be sufficient for Internet2 service creation; scalability and security has to be mainstream and part of the application development life cycle.

This position paper suggests how the Internet2 will look like in the future and what are the measures need to be looked into so that an application can remain secure and scale with a guaranteed *quality of service* (QoS). This paper also introduces tools that address the non-functional requirements. “*Phoenix*” for Scalability and Availability; and “*Suraksha*” for Security.

II. INTERNET2 APPLICATIONS

In our opinion, Internet2 will be a combination of

- Multi-user-agent
- Multi-service
- Multi-access
- Multi-provider
- Multi-protocol networks
- Web 2.0 and Web 3.0
- IPv6 with IPsec
- Support mobility at vehicular state
- Intelligent and programmable networks
- Definable service quality
- Definable security level
- On demand scalability
- API in the network to obtain context information (spatial, environmental, and temporal attributes)
- API in the network to enforce QoS and security

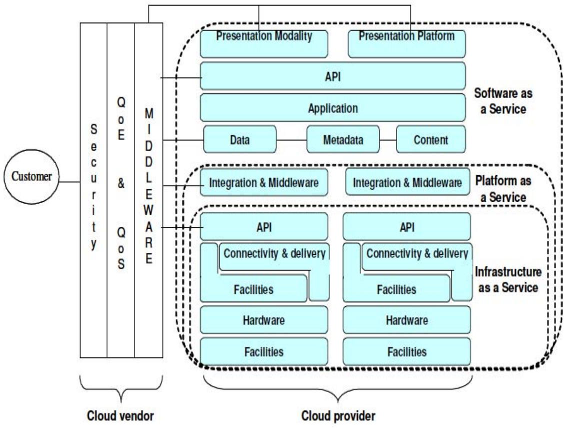

Figure 1: Internet2 Architecture

In our thought, Internet2 deployment architecture will be similar to Figure 1. There will be a *cloud vendor* that offers services to the end user. These services can be peer-to-peer content delivery, social networking services, healthcare/telemedicine services, web-based communities, Web 2.0, Web 3.0, services with application mash-ups, vehicular network applications, context-aware services, you name it. It is possible that the cloud vendor itself offers the SaaS, PaaS, and IaaS. Like the application, the cloud vendor will be agnostic to the content, software, platform, and the infrastructure. The cloud vendor will offer the best possible deployment environment at best possible price. The cloud vendor will ensure security, scalability, interoperability, and availability. The cloud vendor will ensure quality of service through *service level agreement* (SLA).

III. **SCALABILITY AND AVAILABILITY IN INTERNET2**

Scalability in Internet2 applications needs to be dynamic – these applications need to react to external events such as increased load, augmented reliability etc. In Internet2, there will be no association between the service and the underlying platform and infrastructure. The cloud user / customer may like to use some resource somewhere in the world that offers the best price performance advantage – the model telecommunication vendors used for long. Therefore, the cloud vendor will create a virtual execution environment for the service that can leverage this advantage. The scalability in such environment will be managed at two levels. These are platform level (PaaS) and the software level. For scalability the application must be able to leverage,

- Tightly coupled parallelism – here the application need to use instruction-level parallelism at the application layer that uses multi-threading techniques and uses technologies like message passing interface (MPI) [8] or OpenMP [9].
- Loosely coupled parallelism – here the application need to use task-level parallelism at the platform level by using intelligent middleware such as MapReduce [10] and Apache Hadoop [11].
- Cloud level parallelism – A flavor of loosely coupled parallelism paradigm that allows creating software as a service. Platforms such as GridGain [12] can be used to develop and run applications on private or public clouds.

In this paper we try to observe a variety of approaches to design scalable Internet2 applications. In this context we propose a system called Phoenix [13], an intelligent cloud middleware that can react to the dynamic needs of Internet2 based applications. This system can help applications with varied load to scale up or scale down by provisioning virtual environment on-demand. It is a loosely coupled system to hide the complexities of scalability from the applications.

Phoenix can manage infrastructures such as Hadoop and GridGain. While these infrastructures will ensure availability and provide fault tolerance to underlying hardware/network failure, it can rely on Phoenix to gurantee the quality of service.

Phoenix provides centralized management of VM workloads and distributed infrastructure. It supports various VM placement policies and allows dynamic partition and isolation of clusters. Phoenix supports heterogeneous execution environments with multiple, even conflicting, software requirements on the same shared infrastructure and provides full control of the lifecycle of virtualized services management. Phoenix has an open and flexible architecture that allows integrating other open source software. Some of the cardinal features of Phoenix are Virtualization management, Image management, Network management, Fault tolerance, Access control and scalability.

Biologicl and Lifesciences problems are NP-hard; these applications need tremendous coputing power to process terabytes to petabytes of data. As a test-bed we have built *Bio- Cloud* – a next generation cloud application on top of Phoenix that provides a complete and exhaustive downstream bioinformatics and biostatistics analysis of Next Generation Sequencing data. These applications will be used by biologists, research labs, and *small & medium enterprises* (SME) in biotechnology.

IV. **SECURITY IN INTERNET2**

Security in Internet2 needs to answer few specific questions like,

1. How much trust do you have on virtualized environment or the hypervisors in the cloud as against your own physical hardware?
2. How much trust do you have on cloud vendor versus your own infrastructure?
3. How do you address regulatory and compliance requirement in an environment when your application might be running on an infrastructure in a foreign country?

To answer these, security in the Internet2 will need following attributes,

- Confidentiality
- Integrity
- Authentication
- Authorization
- Anonymity

Confidentiality will ensure that the data is encrypted using some of the accepted cryptographic algorithms of public key cryptography like RSA, Elliptic curve; symmetric cryptographic algorithms like AES or 3DES; key exchange algorithms like Diffie-Hellman; and, digital signatures;. Data can be a message, a piece of data, XML, or even HTTP data. For real-time data, even symmetric stream ciphering algorithms like RC4 can be used. Algorithms like MD5 or SHA will also be used to create digital signature for integrity.

Authentication and *Authorization* (A&A) will be a major challenge in Internet2. A&A will be at two major levels – at the user level within the network and at the application level at the computing platform end. In the Internet2 the user will seamlessly roam between networks. These networks can be a wired netwok or wireless network – WiFi or 3G or even WiMAX networks. Along with mobile IP, the network will provide seamless A&A, handoff, and roaming between homogenous and heterogenos networks. To ensure seamless roaming, the network must offer Intradomain & Interdomain A&A and security through 3GPP standards like: • Security Architecture and Authentication and Key Agreement (AKA) [3GPP TS 33.102]

- Network Domain Security (NDS) [3GPP TS 33.310]
- Access Security for SIP-based Services [3GPP TS 33.203]
- Generic Authentication Architecture [3GPP TS 33.220]
- Access Security for HTTP-based Services [3GPP TS 33.222]

At the application level the security will cover elements like Availability, Anonymity, and Object security. Part of the availability will be ensured by fault-tolerance as described in Section 3; rest will be through measures against *denial of service* (DoS) attack. Countermeasures against DoS attack will be handled at the network level through isolation of subnetworks. As we described, in Internet2 objects and software applications will be agnostic to the underlying platform. The deployment vendor must guarantee anonymity so that the data and content is confidential and anonymous. In the virtual environment any platform or infrastructure might be compromised. Therefore, the application must be security aware and need to ensure that security is ensured at various levels, that include,

- Service to Platform security – this category will represent the set of threats in which a compromised services (or a malicious service exploits security weaknesses of a platform or launches attack against a platform.
- Service to Service security – this category represents the set of threats in which one service exploit security weaknesses of other services or launch attacks against other applications or objects.
- Platform to Service security – this category represents the set of threats in which compromise platforms attack services.

In this context we propose a system called Suraksha [14], an Open Source tool that provides developers with powerful and elegant technology to design and develop security aware cloud applications.

V. **SERVICE QUALITY AND CHARGING THE USER**

UML or other standard techniques in application development does not pay much attention towards journaling. Journaling is necessary to record usage history. These usage records are used at a later time to charge and bill a consumer. In addition, Journaling is a critical part of any secured and reliable system; it helps a system to recover from failure through either roll-back or roll-forward. Journaling is also used for forensic purposes to recreate the scene of a security attack. In our opinion, billing and charging in Internet2 will be similar to billing and charging in the Telecom space; however, the spatial (distance between caller and called) attribute of telecom will not play any role in I Internet2. In I Internet2, billing will be a combination of fixed charges, recurring charges, and charges based on usage. In I Internet2, additionally, charging will depend on the *quality of service* (QoS) and *quality of experience* (QoE) attribute driven by *service level agreement* (SLA). SLA will cover both S&S requirements of security & scalability. The SaaS and inter-service provider billing will be complex in Internet2. Inter-service billing will be based on slabs and bill & keep model. Similar models will be applicable at the PaaS and IaaS level. However, SaaS billing will depend not only on the content but also on the *intent*. Monetization in Internet2 will be driven by value and not on size of the software. Many services in the Internet2 will monetize from deriving the intent, emotions, and opinion of the consumer. There are a few open-source tools that can be enhanced by a vendor to manage QoE and charging of services at the platform level –these are OpenNebula, Haizea, and Phoenix.

- OpenNebula [15] is a Virtual Infrastructure Manager that orchestrates storage, network and virtualization technologies to enable the dynamic placement of multi-tier services (groups of interconnected virtual machines) on distributed infrastructures, combining both data center resources and remote cloud resources, according to allocation policies.
- Haizea [16] is an open source virtual machine-based lease management architecture. In combination with the OpenNebula virtual infrastructure manager it can be used to manage a Xen, KVM, VMWare, etc. clusters, allowing one to

deploy different types of leases that are instantiated as virtual machines (VMs).

- Phoenix [13] is an open source cloud middleware used to build, manage and administer on-premise or hybrid cloud. Phoenix is an extension of OpenNebula and is built to be hypervisor agnostic and can leverage schedulers such as Haizea for intelligent lease management of the virtual infrastructure. Phoenix has built in robust Fine Grained Security and Group Management, Storage Provisioning, transaction manager and Intelligent Cloud scale for varied application loads such as *high performance computing* (HPC) or business application to leverage underlying cloud infrastructure.

To ensure security and service quality in NGI, a vendor has to go beyond its own domain of control. This becomes even more complex when the service provider does not own some of these service infrastructures. Therefore, we recommend that a cloud vendor uses ITU-T Recommendation G.1000 [17] for QoS. An SLA is a contract between a customer and the vendor to define QoS – to ensure *quality of experience* (QoE); it can be implemented in the NGI using definitions and rules [18]. In case the terms of the SLA contract are violated [19] the vendor must ensure recovery and corrective actions.

VI. CONCLUSION

Internet2 have made significant irreversible changes in the way people use Internet and create services developing and testing new technologies, such as IPv6, multicasting and quality of service (QoS) that will enable revolutionary Internet applications. However, these applications require performance not possible on today's Internet. More than a faster Web or email, these new technologies will enable completely new applications such as digital libraries, virtual laboratories, distance-independent learning and tele-immersion. A primary goal of Internet2 is to ensure the transfer of new network technology and applications to the broader education and networking community applications, services, and infrastructure in the past were designed to meet the need in deterministic terms. Unlike the static environments for services and applications deployment, the deployment scenario for Internet2 will be dynamic – user expectation, user volume, and demand on infrastructure will be dynamic. Also, Internet2 will be available to every individual starting from common users to hackers. This will make the availability, anonymity, and security challenges much more complex. In this position paper, we presented these challenges and proposed techniques to mitigate them.

REFERENCES

[1] Rumbaugh J, Jacobson I, Booch G, The Unified Modeling Language Reference Manual, Addison-Wesley, 1999.

[2] Asoke K Talukder and Manish Chaitanya, Architecting Secure Software Systems, Auerbach Publications, 2008.

[3] Guttorm Sindre and Andreas L Opdahl, "Capturing Security Requirements by Misuse Cases," in Proc. 14th Norwegian Informatics Conference (NIK'2001),Troms, Norway, Nov. 2001.

[4] G. Sindre and A.L. Opdahl, "Eliciting Security Requirements by Misuse Cases," in Proc. 37th Conf. Techniques of Object-Oriented Languages and Systems, TOOLS Pacific 2000, 2000, pp. 120–131.

[5] G. Sindre and A.L. Opdahl, "Eliciting security requirements with misuse cases," Requirements EInternet2neering, Vol. 10, No. 1, pp. 34-44, Jan.2005.

[6] Talukder, Asoke K.; Maurya, Vineet Kumar; Santhosh, Babu G.; Jangam, Ebenezer; Muni, Sekhar V.; Jevitha, K. P.; Saurabh, Samanta; Pais, Alwyn Roshan, "Security-aware Software Development Life Cycle (SaSDLC) - Processes and tools," Wireless and Optical Communications Networks, 2009. WOCN '09. IFIP International Conference on , vol., no., pp.1-5, 28-30 April 2009.

[7] Wikipedia – the free encyclopedia – www.wikipedia.org.

[8] The Message Passing Interface (MPI) standard http://www.mcs.anl.gov/research/projects/m i/

[9] The OpenMP API specification for parallel programming http://openmp.org/wp/ [10] MapReduce: Simplified Data Processing on Large Clusters http://labs.google.com/papers/mapreduce.html

[11] The Apache Hadoop project develops open-source software for reliable, scalable, distributed computing - http://hadoop.apache.org/

[12] Cloud development Platform - www.gridgain.com

[13] Phoenix: Open Source On-Premise Cloud Eco system www. geschickten.com/RD.html

[14] Suraksha: Open Source platform of developing security aware cloud applications- www.geschickten.com/RD.html

[15] OpenNebula, a Virtual Infrastructure Manager http://www.opennebula.org/doku.php

[16] Open Source virtual machine-based lease management architecture http://haizea.cs.uchicago.edu/

[17] ITU-T Recommendation G.1000, Communications quality of service: A framework and definitions.

[18] RFC3644 - Policy Quality of Service (QoS) Information Model.

[19] Antony Oodan, Keith Ward, Catherine Savolaine, Mahmoud Daneshmand, Peter Hoath, Telecommunications Quality of Service Management: From Legacy to Emerging Services, Institution of Electrical Engineers, 2002.