Security Management of Cloud-Native Applications

Presented By:
Rohit Sharma
MSc in Dependable Software Systems (DESEM)

Outline

- Context
- State-of-the-Art
  - Design Patterns
  - Threats to cloud systems
  - Security Mechanisms
- Contribution
- Evaluation
- Preliminary Results
- Conclusion and Future Work

Context

- **Cloud Computing:** A model for enabling ubiquitous, convenient, on-demand network access to shared pool of configurable computing resources that can be *rapidly provisioned* and released *with minimal management effort* or service provider interaction. [NIST2011]

- **Cloud-Native Application:** Computer software that natively utilizes services and infrastructure provided by cloud service providers.

Context

Overall Cloud Computing Concerns

- **Network Availability**: Cloud must be available whenever you need it.
- **Disaster Recovery & Business Continuity**: Services should continue even if cloud provider's production environment is subject to disaster.
- **<u>Security Concerns</u>**
- **Data Isolation**: Tenant's confidential data may be comingled with data belonging to others.
- **Security Incidents**: Tenants need to be informed of security incidents. Also, tenants may require provider support for audits.
- **Virtualization based Risks**: Heavy use of virtualization introduce new risks such as attacks among VMs on same physical server.
- **Data Security**: Storing unencrypted data on the cloud can be risky.

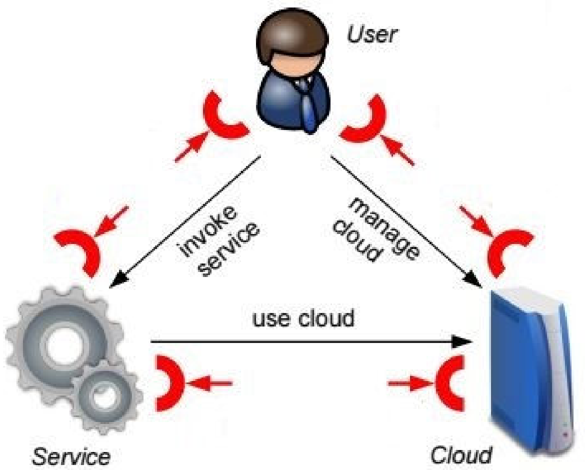
State of the Art

Cloud-Native Applications

- **Properties of Cloud-Native Application:**
  - Distribution
  - Elasticity
  - Isolated State
  - Automated Management
  - Loose coupling
  - Resiliency
- **Design Patterns**
  - Elasticity Manager
  - Loose Coupling
  - Stateless Components
  - Stateful Components
  - Message Mover
  - Resiliency

State of the Art

Threats to cloud systems



- Classification of Threats
  - Based on attack surfaces [Gruschka, Jensen2010]
  - Based on layers [C. Modi et al.2012]
- Attack Surfaces
  - User to Service/ Service to User
    - (Buffer-overflow/ SSL certificate spoofing)
  - Service to Cloud/ Cloud to Service
    - (Resource Exhaustion Attacks / Rootkits)
  - User to Cloud/ Cloud to User
    - (Attacks on cloud management interfaces / Attacks originating at cloud)
- Based on Layers
  - Data Storage Level (Co-mingling, weak encryption)
  - Network Level (DDOS, IP Spoofing)
  - Virtualization level (Hypervisor Compromise e.g. blue pill)
  - Application Level (SQL injection, XSS)

Threats to cloud systems

Top threats as identified by ***Cloud Security Alliance*** [CSA2010]

- Abuse and Nefarious Use of Cloud Computing (DDOS)
- Insecure Application Programming Interfaces (Reusable tokens)
- Malicious Insiders
- Shared Technology Vulnerabilities (Hypervisor compromise)
- Data Loss/Leakage (Weak authentication, authorization, audit controls)
- Account/ Service/ Traffic Hijacking (Phishing, Exploiting vulnerabilities)
- Unknown Risk Profile (No transparency in internal security procedures, patching, auditing & logging)

State of the Art

Security Mechanisms

- Implementing security controls in *layered* fashion
- **Honeypots**: Create a false non-production system to entice the attacker. Once attacked, distract the attacker and report the incidence
- **Sandboxing**: Add a layer between code and OS. E.g.: Hypervisor
- **Isolation**:
  - Use encryption for VM network traffic to provide logical isolation
  - Isolation using subnets
- **Auditing & Monitoring**
  - Use of Configuration management database (CMDB)
  - Use of Attack signatures to match events of interest
  - Provide feedback loop to the system operating the cloud infrastructure
  - More sophisticated detection possible using situational awareness

Limitations of Security Mechanisms

- Traditional security techniques being applied in cloud infrastructure.
- The techniques not specific to cloud-native applications.
- Security in cloud-native applications is complex as it’s a conjunction of many other services provided by IaaS, PaaS, SaaS.

Contribution

- Goal
  - Build a solution that is capable to address specific security requirements of cloud-native applications
  - Develop a security approach for cloud-native application
- Security Approach for Cloud-Native Applications
  - Cloud-native application modeling
  - Detection Mechanism
  - Prototyping & Evaluation

Contribution

Modeling the Cloud-Native application

- Each software component performs a particular class of tasks
  - (e.g. {Presentation, Authentication, DataAccess}) .
- Each component may be dependent on other software components.
- Each component is elastically scalable
  - Each component may contain multiple instances of same software module
  - Each instance is represented by a node
  - Each component performs only one class of tasks
- Set **S** = set of nodes *(represents instances of software modules)*
- Set **C** = set of class of tasks *(can be mapped to functional clusters)*
  - Function *Role: $S \rightarrow C$* associate nodes with functional clusters
- Each cluster may be dependent on other clusters
  - Communication among clusters can be represented by $\boldsymbol{\eta \subseteq C \times C}$

Contribution
Abstract Model

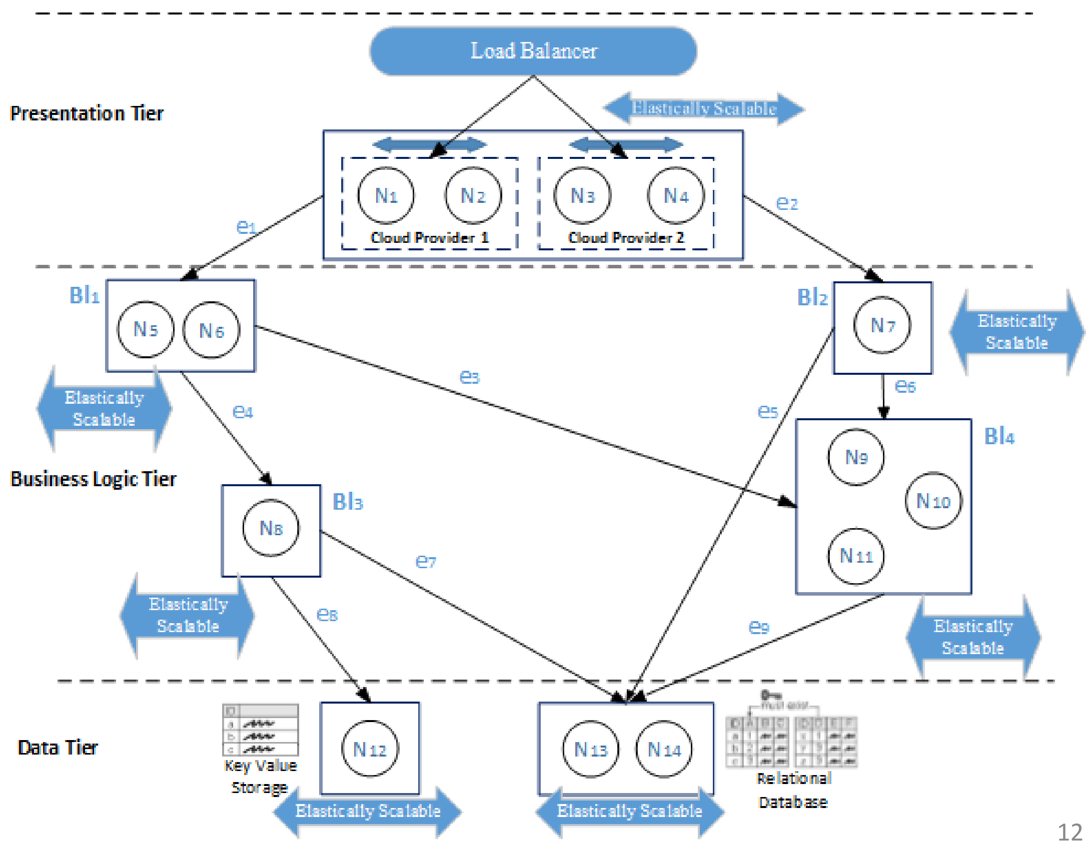
- Nodes **S** =

  {$N_1, N_2, ... N_{14}$ }
- Clusters **C** =

  { *Presentation,*

  Bl_1, Bl_2, Bl_3, Bl_4,
  *KeyValue, RelationalDB***}**
- Dependencies **η** =
  - {$e_1, e_2, ..., e_9$ }
- Cloud Application **App**=

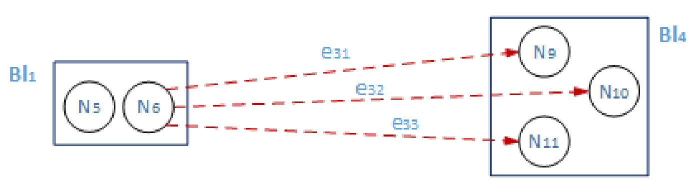
  <C, η> where η ⊆ C x C



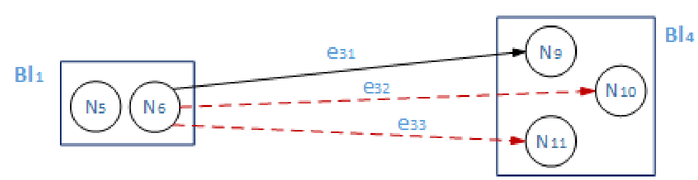
Contribution

Communication Among Nodes

- Abstract model does not specify communication among specific instances.
- Only one edge per request between one cluster to another.
- Elastic load balancer decides the instance to which the request should be forwarded within a cluster.
- We mimic this behavior using notion of *probable edges* out of which only one edge (*selected non-deterministically*) is *realized.*



Probable edges



Realized edge

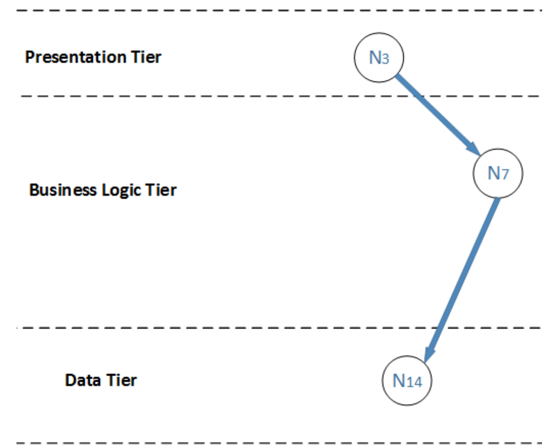
Contribution

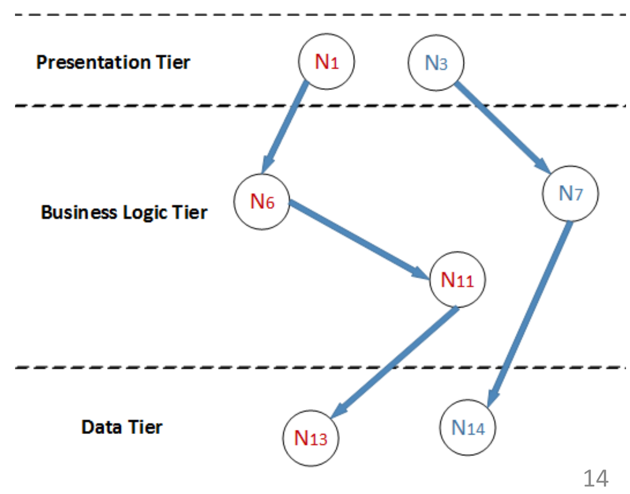
Modeling the Computation

- Set of requests **R** = $\{r_1, r_2, \ldots\ldots\ldots, r_n\}$
- Each request $\mathbf{r_i} \in \mathbf{R}$ can be seen as interaction among a set of nodes

  $S' = \{N_i\} \in S$
- Each interacting node shall belong to a different cluster

  $\forall N_i \in S'$, *Role*(N_i) is distinct
- Each request processing can be represented as a graph
  - $\mathbf{G_i} = \langle S', \eta' \rangle$ where $\eta' \subseteq S' \times S'$ *(realized edges)*





Contribution

Detection Mechanisms

- We want to analyze the behavior of cloud-native applications.
- To detect potential attacks by analyzing abnormal behavior.
- We use a detection approach based on k-means clustering to detect malicious activities.

Contribution

Some Possible Symptoms

- Request path not complete
  - Request trace did not reach one of the nodes in a set of termination nodes.
  - **Possible Reasons:** Low level exploits such as **EIP overwrite**, crashes due to stack overflow.
- Deviation from dependency path
  - Each cluster may be dependent on other clusters.
  - Abnormal behavior in case the request path violates the dependency path.
  - **Possible Reasons**: Vulnerabilities in APIs.
- Biasness
  - Some nodes may be biased towards a particular dependent cluster.
  - **Possible Reasons**: Misuse of a particular feature in the application .

Evaluation

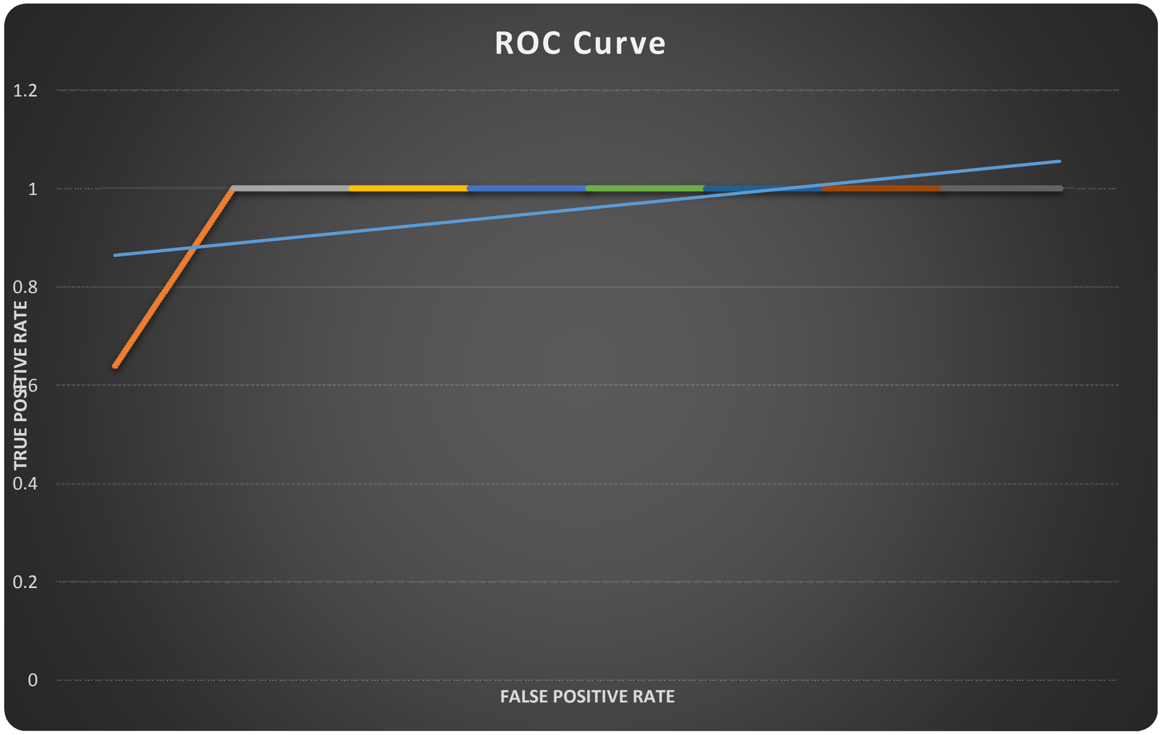
- Built a simulator to represent the behavior of cloud-native applications and potential attacks.
- Using k-means clustering algorithm to detect potential attacks.

Evaluation

Preliminary Results

- Receiver operating characteristic (ROC) curve generated using *sensitivity* and *specificity* calculations.

- Allows to evaluate the performance of the detection
- The current performance is good but we have modeled very simple attacks



Conclusions & Future Work

- Proposed Security Approach for Cloud-Native Applications
  - Cloud-native application modeling
  - Detection Mechanism
  - Prototyping & Evaluation

- Future work
  - Modeling more complex attacks.
  - Extracting traces from openstack infrastructure.

Questions ?