CERN

OpenStack Heat @ CERN

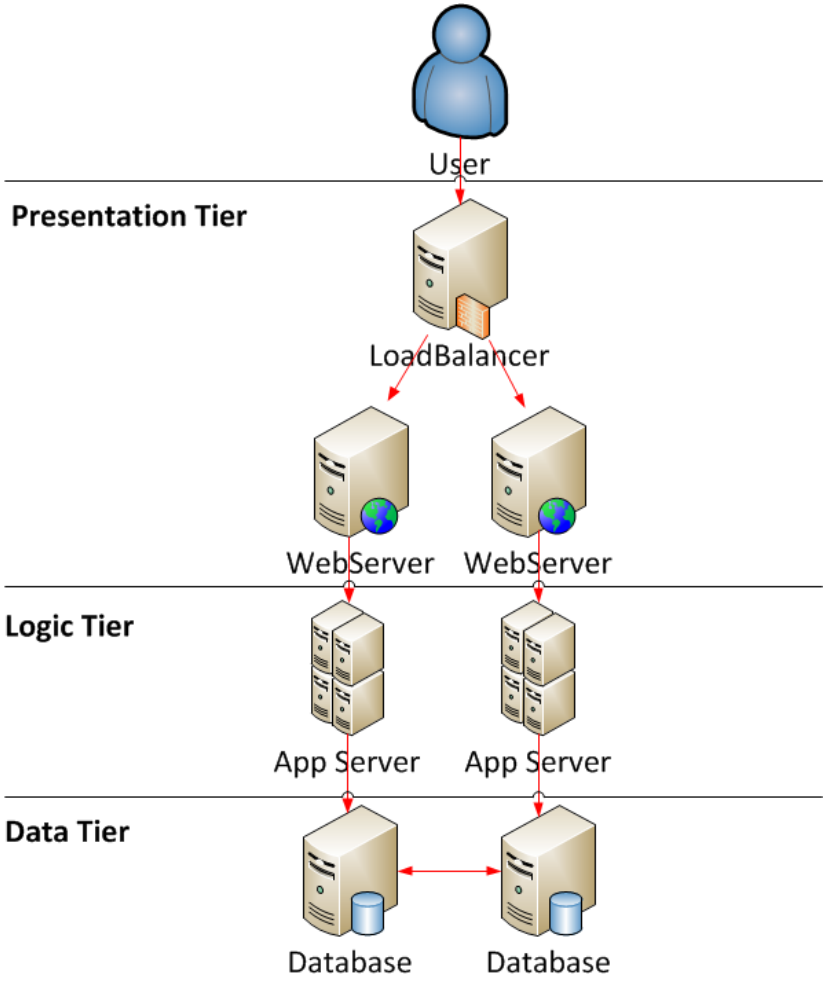
Bruno Bompastor (bruno.bompastor@cern.ch)
CERN Cloud Team

HEPiX Spring 2015
Oxford University, UK

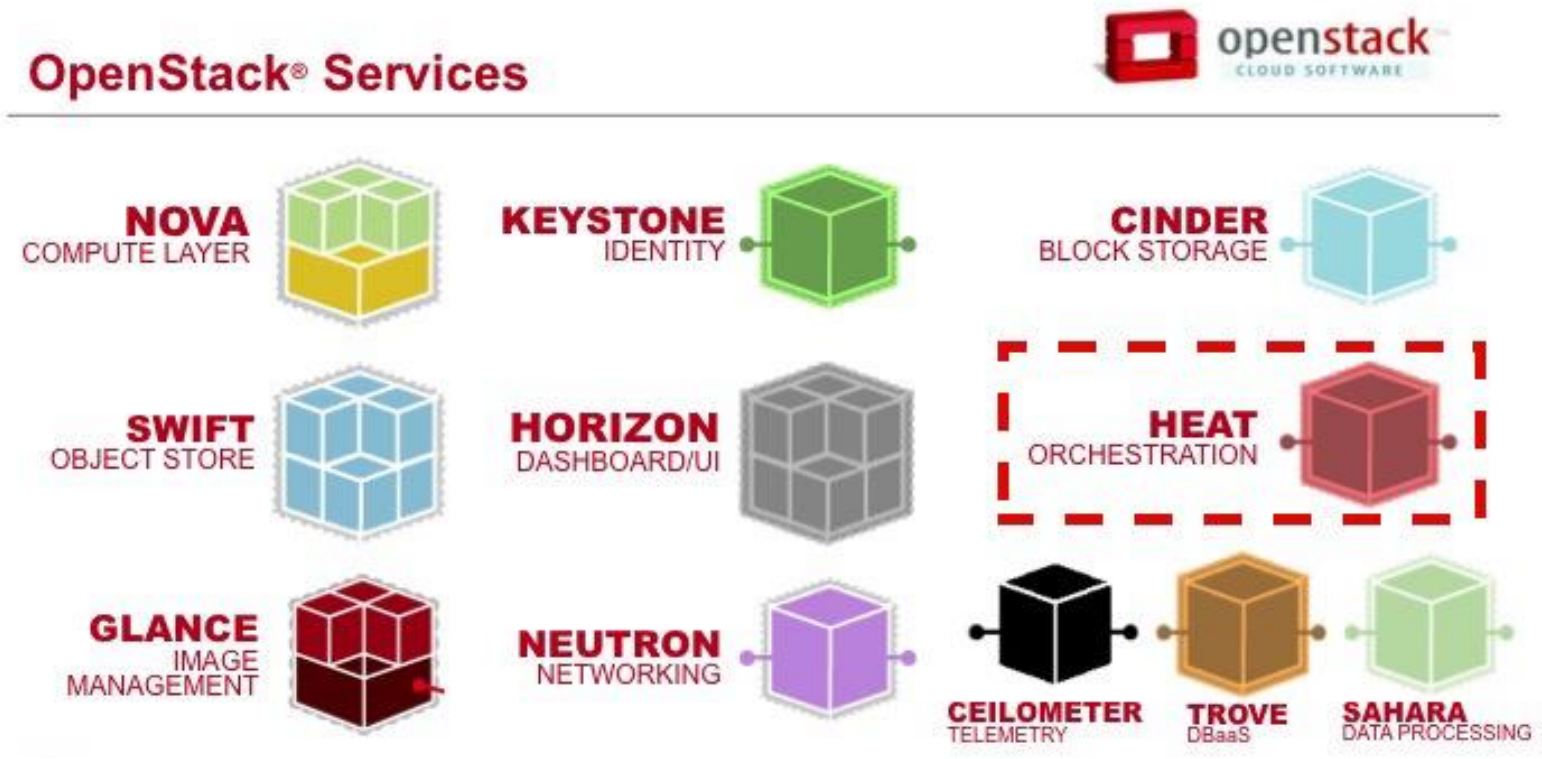
Outline

- Motivation
  - What is Heat?
  - Architecture Overview
  - Stack Concept
- Heat
  - Problems and Solutions
- Use Cases @ CERN
- Deployment
- Future Plans

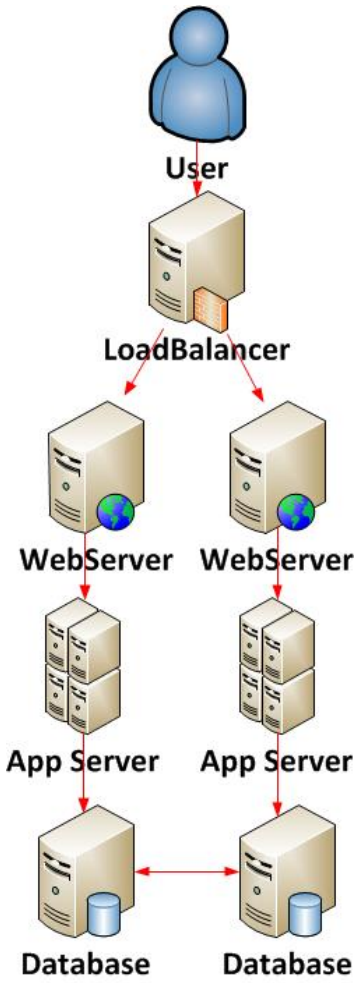
3-Tier Application



OpenStack Overview



Problem 1: Create an Application



LoadBalancers:

- nova boot lb01 --flavor m1.small –image lb-cc7

WebServers:

- nova boot ws01 --flavor m1.medium –image ws-cc7
- nova boot ws02 --flavor m1.medium –image ws-cc7

App Servers:

- nova boot ap01 --flavor m1.medium –image ap-cc7
- nova boot ap02 --flavor m1.medium –image ap-cc7

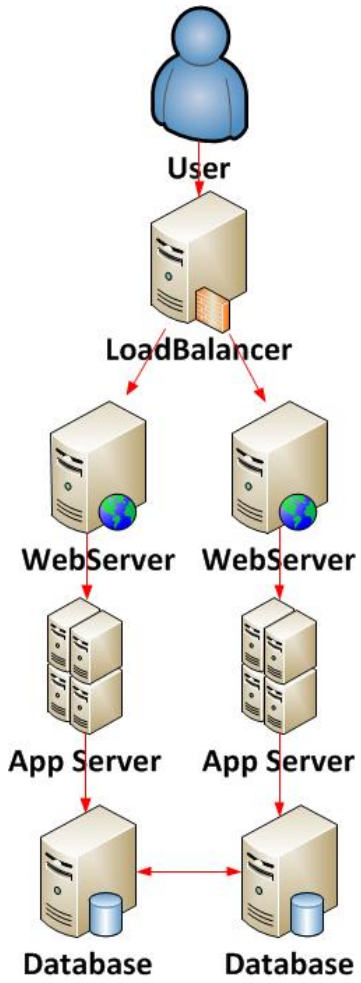
Databases:

- nova boot db01 --flavor m1.large –image db-cc7
- nova boot db02 --flavor m1.large –image db-cc7

Optional:

- cinder create --volume_type standard --display_name vol01 150
- cinder create --volume_type standard --display_name vol02 150
- nova volume-attach db01 vol01 auto
- nova volume-attach db02 vol02 auto

Solution 1: Template



Application:

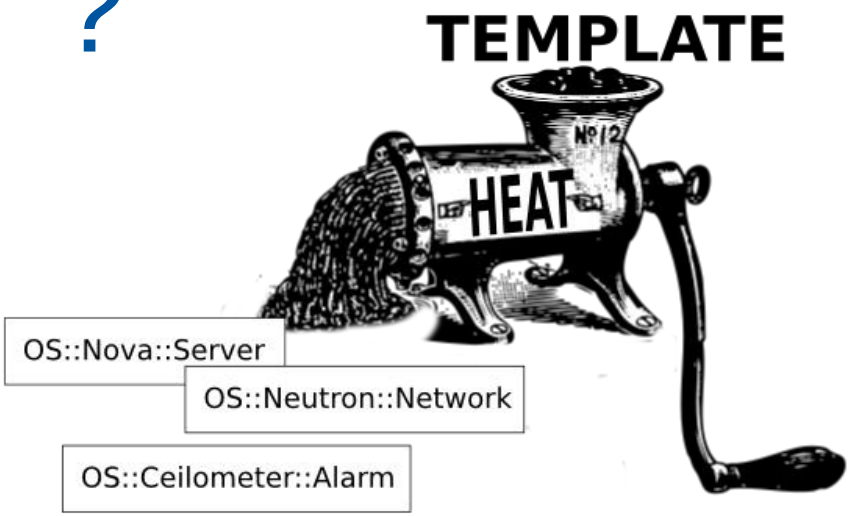
- **heat stack-create app01 -f app01.yaml**

Snippet of Heat Template (app01.yaml):

```
resources:
 loadbalancer01:
  type: OS::Nova::Server
   properties:
    name: loadbalancer01
    image: lb-cc7
    flavor: m1.small
```

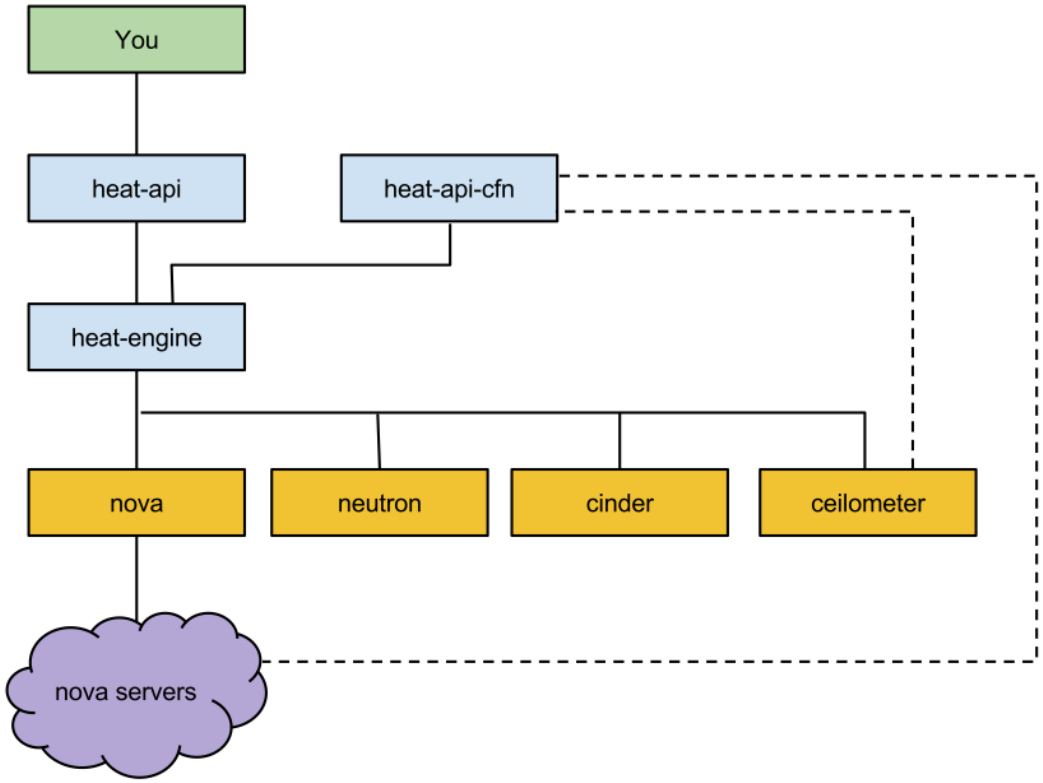
```
webserver01:
 type: OS::Nova::Server
  properties:
   name: webserver01
   image: ws-cc7
   flavor: m1.medium
```

What is heat ?



- Heat provides a mechanism for orchestrating OpenStack resources through templates
  - Analogous to AWS cloud formation
  - Re-use of AWS cloud formation templates

Architecture Overview

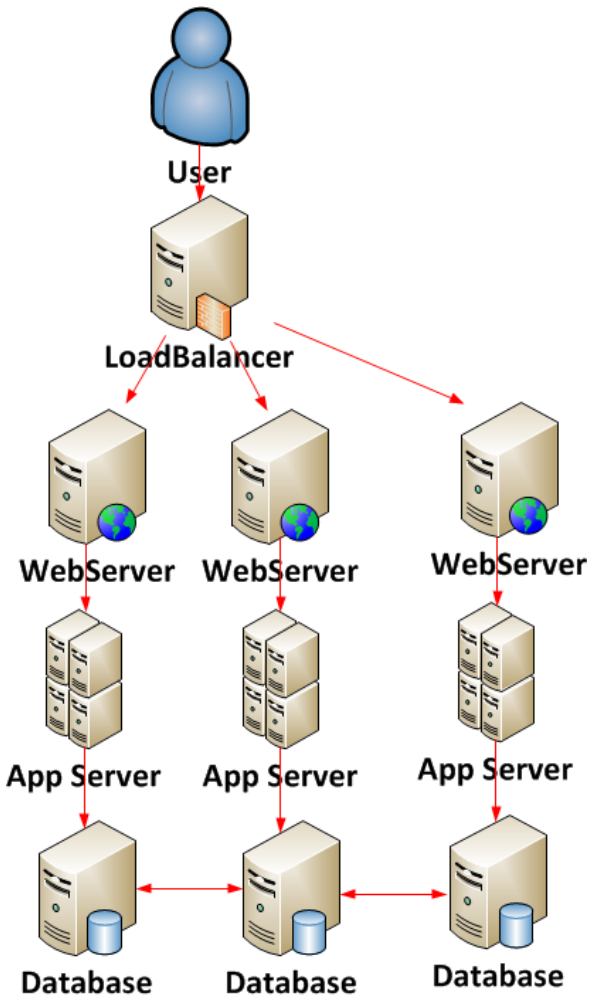


Stack Concept

Stack Resources

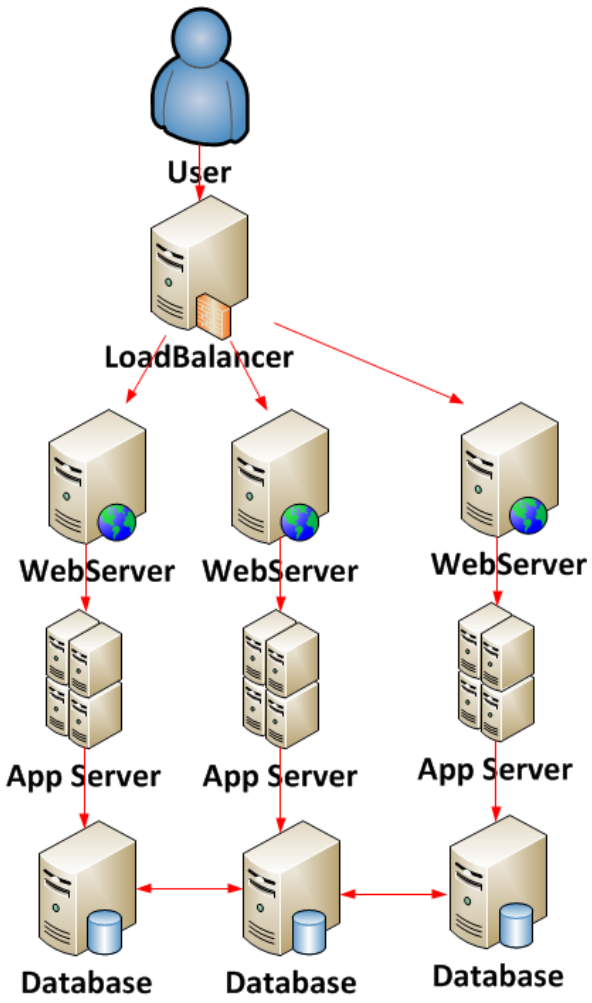
Stack Resource	Resource	Stack Resource Type	Date Updated	Status	Status Reason
application02	f7824b16-5a4a-4412-bd3e-9b297ea14007	OS::Nova::Server	1 minute	Create Complete	state changed
application01	e920d954-8d9d-4bc8-b6f1-72c4c43d69aa	OS::Nova::Server	1 minute	Create Complete	state changed
database01	93f23566-cd1d-44d4-a462-8b8e4162d27e	OS::Nova::Server	1 minute	Create Complete	state changed
database02	90488460-a85f-4851-9e87-ad1d5b5119e3	OS::Nova::Server	1 minute	Create Complete	state changed
webserver01	8fdaebf8-3e5f-421d-b47a-d508cc31b6ea	OS::Nova::Server	1 minute	Create Complete	state changed
loabalancer01	6a456a11-4a30-43c2-9e6f-fc92bc98ff20	OS::Nova::Server	1 minute	Create Complete	state changed
webserver02	eb72a0b6-853a-4c15-a3aa-dc59d78c271e	OS::Nova::Server	1 minute	Create Complete	state changed

Problem 2 – Grow the Application



- Problem:
  - Need for a service/application to grow
  - Way to automate the creation of resources
- Solution:
  - Heat templates allows to automate the creation of OpenStack resources

Solution 2 – Resource Group



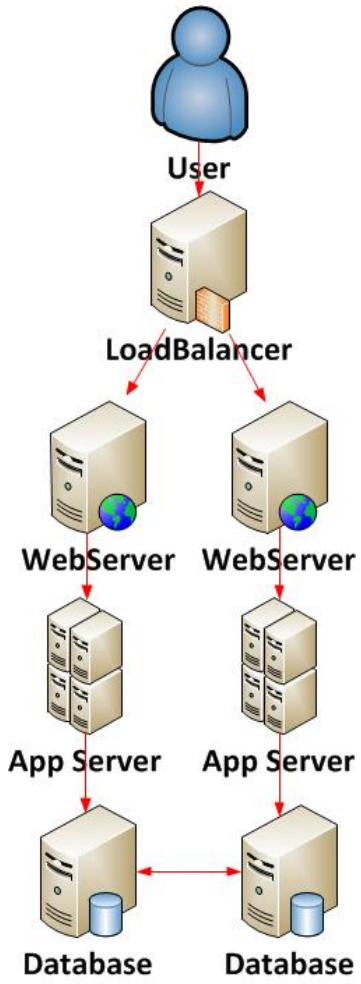
Create Web Server (webserver.yaml):

```
ws:
    type: OS::Nova::Server
    properties:
      image: ws-cc7
      flavor: m1.medium
```

Increase # of web servers (app01.yaml):

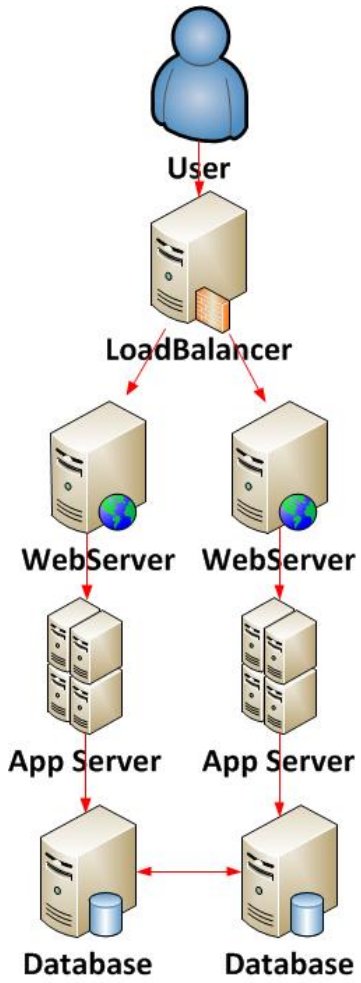
```
resources:
  rg:
    type: OS::Heat::ResourceGroup
    properties:
      count: 3
      resource_def: {type: webserver.yaml}
```

Problem 3 – Configure the App



- Problem:
  - Configure application servers (web server, database, etc)
  - Way to automate this configuration
- Solution:
  - Heat templates allows to insert user data via cloud-init

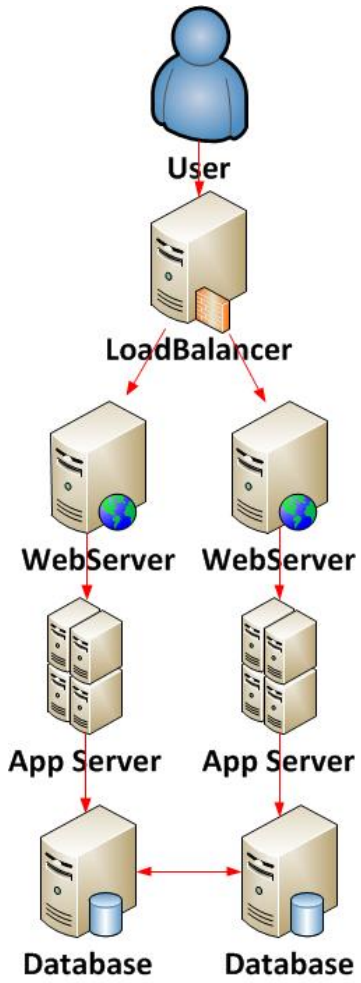
Solution 3 – Template User Data



Configure the Web Server via user data:

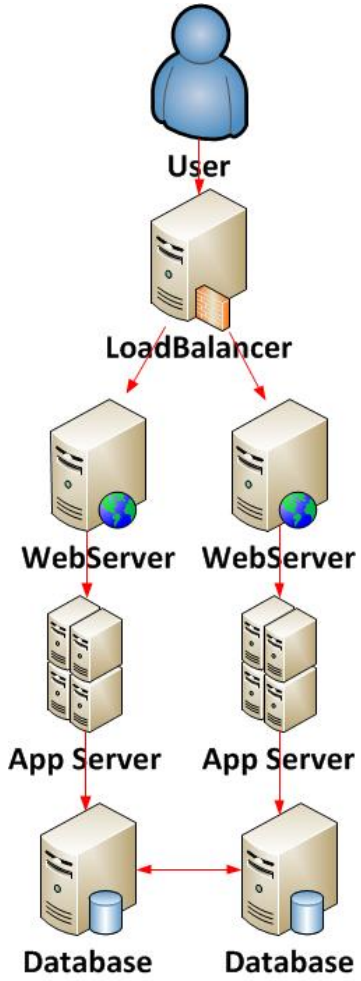
```
webserver01:
  type: OS::Nova::Server
  properties:
    image: base-cc7
    flavor: m1.medium
    user_data_format: RAW
    user_data:
      str_replace:
        template: |
          #!/bin/sh
          yum install -y httpd
          service httpd start
          iptables -I INPUT 4 -m state --state NEW -p tcp --dport 80 -j
ACCEPT
          service iptables save
          service iptables restart
```

Problem 4 – Startup Order



- Problem:
  - Steps to configure an application:
    1. Database
    2. Application Server
    3. Web Server
    4. Load Balancer
- Solution:
  - Heat mechanism to notify when a resource has finished all its operations
  - Uses a web hook (via curl or cfn-signal) to notify heat of an event
  - The notification mechanism coupled together with a dependency on the resource allows to make a startup order

Solution 4 – Notifications and Dependencies



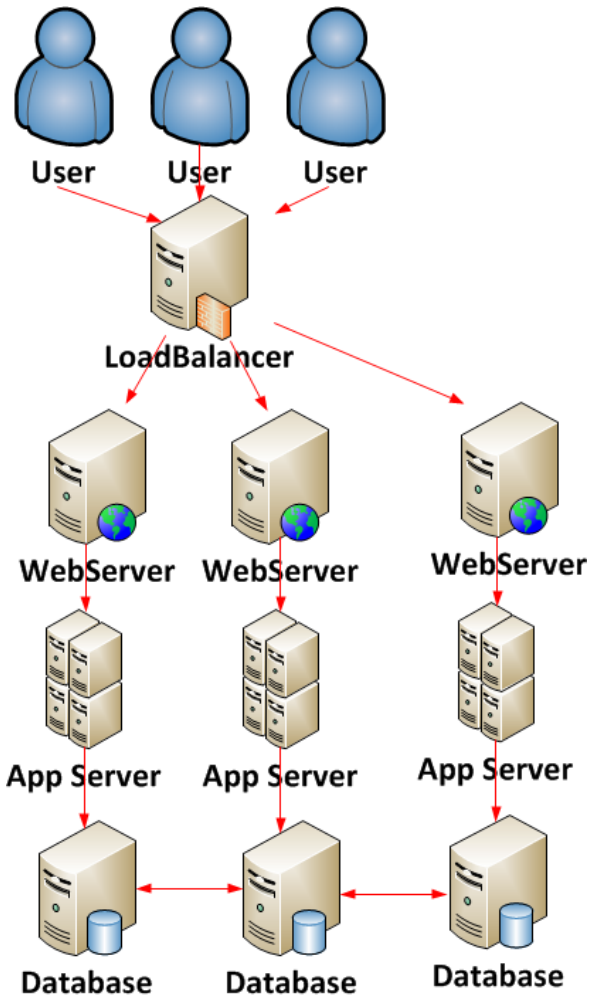
Web Server depends on Application Server:

```
webserver01:
  type: OS::Nova::Server
  depends_on: application01
  properties:
    name: webserver01
    image: ws-cc7
    flavor: m1.medium
```

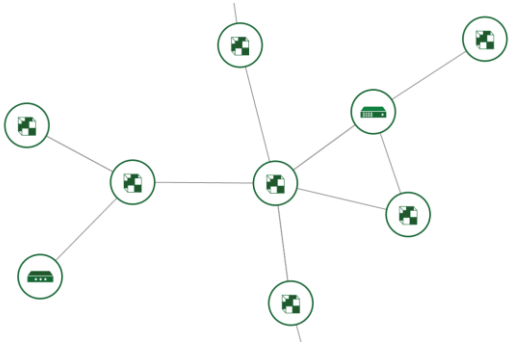
Notify Heat that application server finished:

```
user_data:
    str_replace:
      template: |
        #!/bin/bash
        # Signal heat that we are finished settings things up.
        wc_notify --data-binary '{"status": "SUCCESS"}'
      params:
        # Create the curl command
        wc_notify: { get_attr: ['wait_handle', 'curl_cli'] }
```

Problem 5 – Scale the Application



- Problem:
  - More users utilizing the application
  - Need for more resources to coupe with the increase of requests
- Solution:
  - Heat allows to build a stack that automatically reacts to events through time
  - Uses ceilometer alarms to monitor resources



Solution 5 – AutoScaling

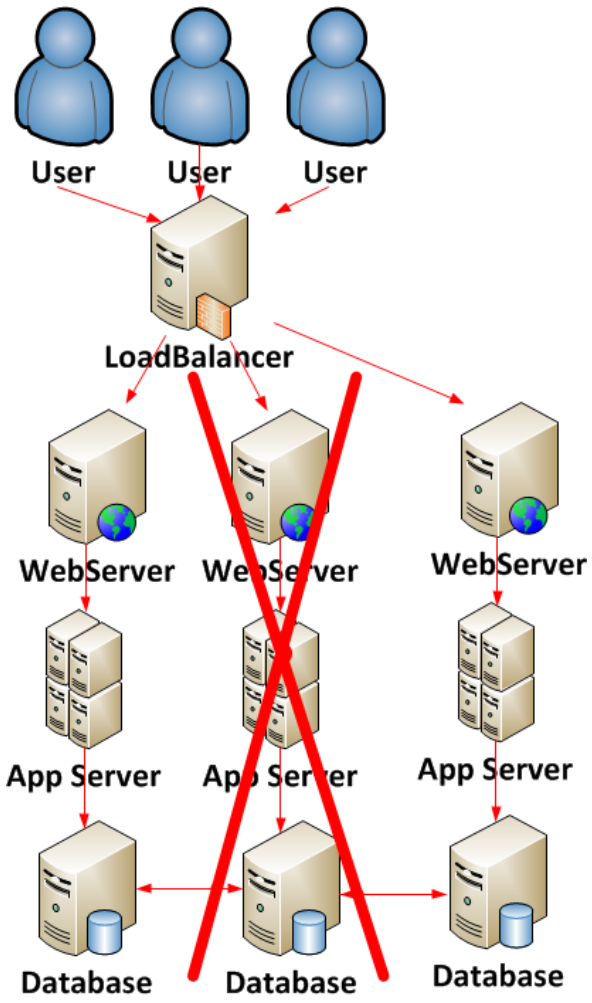
Create ceilometer alarm for CPU utilization:

```
cpu_alarm_high:
  type: OS::Ceilometer::Alarm
  properties:
    description: Scale-up if the average CPU > 50% for 1 minute
    meter_name: cpu_util
    statistic: avg
    period: 60
    evaluation_periods: 1
    threshold: 50
    alarm_actions:
      - {get_attr: [web_server_scaleup_policy, alarm_url]}
    matching_metadata: {'metadata.user_metadata.stack': {get_param: "OS::stack_id"}}
    comparison_operator: gt
```

Create heat scale up policy:

```
web_server_scaleup_policy:
  type: OS::Heat::ScalingPolicy
  properties:
    adjustment_type: change_in_capacity
    auto_scaling_group_id: {get_resource: web_server_group}
    cooldown: 60
    scaling_adjustment: 1
```

Problem 6 – React to Failures



- Problem:
  - More users utilizing the application
  - Resources not able to react to service usage leading to failure
- Solution:
  - Heat allows to define a mechanism for healing resources
  - Alarms trigger creation and deletion of faulty resource:
    - Ceilometer alarm
    - Heat-cfntools enables monitoring inside the VM
    - Use a simple web hook via curl

Solution 6 – Healing

Create the healing mechanism:

```
serverRestarter:
  type: OS::Heat::HARestarter
  properties:
    InstanceId: {Ref: server}
```

Create the ceilometer alarm that triggers the healing mechanism:

```
  type: OS::Ceilometer::Alarm
  properties:
    description: Scale-up if the average CPU > 50% for 1 minute
    meter_name: cpu_util
    statistic: avg
    period: 60
    evaluation_periods: 1
    threshold: 50
    alarm_actions:
      - {get_attr: [serverRestarter, AlarmUrl]}
    matching_metadata: {'metadata.user_metadata.stack': {get_param: "OS::stack_id"}}
    comparison_operator: gt
```

Custom Plugins

- Allows service providers to extend the capabilities of the orchestration service by writing their own resource plugins
- Makes the integration with the underneath infrastructure easier
- Created CERN-specific plugins:
  - Heat resource to register a web site with SSO
  - Puppet integration (work in progress) – alternative to user data input

Use Cases @ CERN

- Jenkins PaaS
  - Automate the creation of master and slave Jenkins servers
- CMS Tier-0 reconstruction
  - Make sure that OpenStack quota is always being maximized
- Elastic Search Cluster
  - Automate the configuration of elastic search cluster managed by the monitoring team
- Video Conference Servers
  - Avoid big interruptions on the service by using the healing mechanism
- Webcast Cluster
  - Autoscale servers based on usage
- Batch Project
  - Automate the resource creation via template

Deployment

- Current Status
  - Heat test environment connected to production infrastructure available upon request
- Plan
  - OpenStack Juno release will allow us to enable some features not yet deployed (auto-scaling, healing)

Future Plans

- Work with CERN community to integrate Heat with their services
- Explore some High Availability features
  - Integration with OpenStack Neutron makes it easier to create load balancing solutions
- Expand Auto Scaling use cases

Thank you!

Questions?