Config Management & Orchestration

Monitoring

Extending

Examples

The End

Spoiler

Icinga2 : Beyond Monitoring

More than monitoring and metrics

Toshaan Bharvani - VanTosh bvba

<toshaan@vantosh.com>

OSAD Miünich 2019

16 October 2019

$ whoami

Toshaan Bharvani - VanTosh

- From Antwerp, Belgium
- Self-employed engineer, trainer, consultant, http://www.vantosh.com
- Involved with Enterprise OS : PEL, RHEL, CentOS, IBM AIX, OpenBSD, FreeBSD, SLES, ...
- Likes to keep everything secure : SELinux, WebSec, ...
- Lives in a virtual world : KVM, Xen, LXC, PowerVM, z/VM, ...
- Likes automation CfgMgmt / DevOps : Ansible, Foreman, Puppet, Mosquitto
- Works on both hardware and software side
- Wants to take over the universe
- Twitter : @toshywoshy
- Blog : http://www.toshaan.com

Table of contents

Config Management & Orchestration

Monitoring

Extending

Examples

The End

Spoiler

1. Config Management & Orchestration
2. Monitoring
3. Extending
4. Examples

Config Management & Orchestration

Monitoring

Extending

Examples

The End

Spoiler

1

Config Management & Orchestration

Ansible

- Orchestration / CfgMgmt / Remote Execution Tool
- Python based uses SSH, WinRM, or API
- Server based, agentless[1]
- host information in flat files, CMDB, scripts, ...
- executes the task on the host side
- Playbook : combination of tasks with meta information
- Roles : abstract playbook with tasks, variables and handlers
- Templates : Jinja2
- works where Python works[2]

[1] Abstraction of SSH / WinRM to connect with the node and in certain setup an agent might be required, but by default it is not necessary

[2] Ansible can actually run without Python on the remote host, however it is not fully supported

Foreman

Foreman is a complete lifecycle management tool for physical and virtual servers. We give system administrators the power to easily autom ¡oy applications, and proactively manage servers, on-premise or in the cloud.

- Automate, discover, provision and upgrade infrastructure
- Ruby on Rails
- Can integrate with Puppet, Chef, Salt, Ansible, . . .

2

Monitoring

Icinga2

Monitoring

Graphing

Dashboard

Alert

Notifications

Monitoring

- Checks the *current* **state** of a subject
- Checks the *current* **utilization** of a subject
- Checks the *current* **return** of a subject

Metrics

- Measures and keeps the *history* of the **states** of a subject
- Measures and keeps the *history* of the **utilization** of a subject
- Measures and keeps the *history* of the **return** of a subject

Hosts & Services

- Hosts
  - Bare Metal Machine
  - Virtual Machine
- Services
  - Application
  - Database
  - State or return value
- Process
  - Database Cluster
  - Web Cluster
  - Loadbalancers

Notifications & Alerts

- Notification
  - Informataion of problem
  - Self remediation
- Alert
  - Failure
  - End user impact

Config Management & Orchestration

Monitoring

Extending

Examples

The End

Spoiler

3

Extending

Icinga2 Extentions

```
object NotificationCommand "service-notification-playbook" {
  import "plugin-notification-command"
  command = [ SysconfDir + "/icinga2/scripts/run-ansible-playbook.sh" ]
  env = {
        NOTIFICATIONTYPE = "$notification.type$"
        SERVICEDESC = "$service.name$"
        HOSTALIAS = "$host.display_name$"
        HOSTADDRESS = "$address$"
        SERVICESTATE = "$service.state$"
        LONGDATETIME = "$icinga.long_date_time$"
        SERVICEOUTPUT = "$service.output$"
        NOTIFICATIONAUTHORNAME = "$notification.author$"
        NOTIFICATIONCOMMENT = "$notification.comment$"
        HOSTDISPLAYNAME = "$host.display_name$"
        SERVICEDISPLAYNAME = "$service.display_name$"
        USEREMAIL = "$user.email$"
  }
}

apply Notification "host-notification" to Host {
  command = "service-notification-mail"
  users = [ "user1", "user2" ]
  period = "24x7"
  states = [ OK, Warning, Critical, Unknown ]
  types = [ Problem,  Recovery ]
  assign where host.vars.notify_host == true
}

apply Notification "service-notification" to Service {
  command = "service-notification-mail"
  users = [ "user1", "user3" ]
  period = "24x7"
  states = [ OK, Warning, Critical, Unknown ]
  types = [ Problem, Acknowledgement, Recovery ]
  assign where service.vars.notify_service == true
}
```

Ansible Extentions

- Icinga2 connects to Ansible Bastion to run playbook
- The Icinga2 ssh user can execute playbooks
-
```
1 ansible-playbook ~/.ansible/playbook/repairmariadbcluster.yml -e 'host=db3'
```
-
```
1 ansible-playbook ~/.ansible/playbook/vm-install.yml -e 'host=newvm0'
```
-
```
1 ansible-playbook ~/.ansible/playbook/repairloadbalancer.yml -l lbweb8
```

Config Management & Orchestration

Monitoring

Extending

Examples

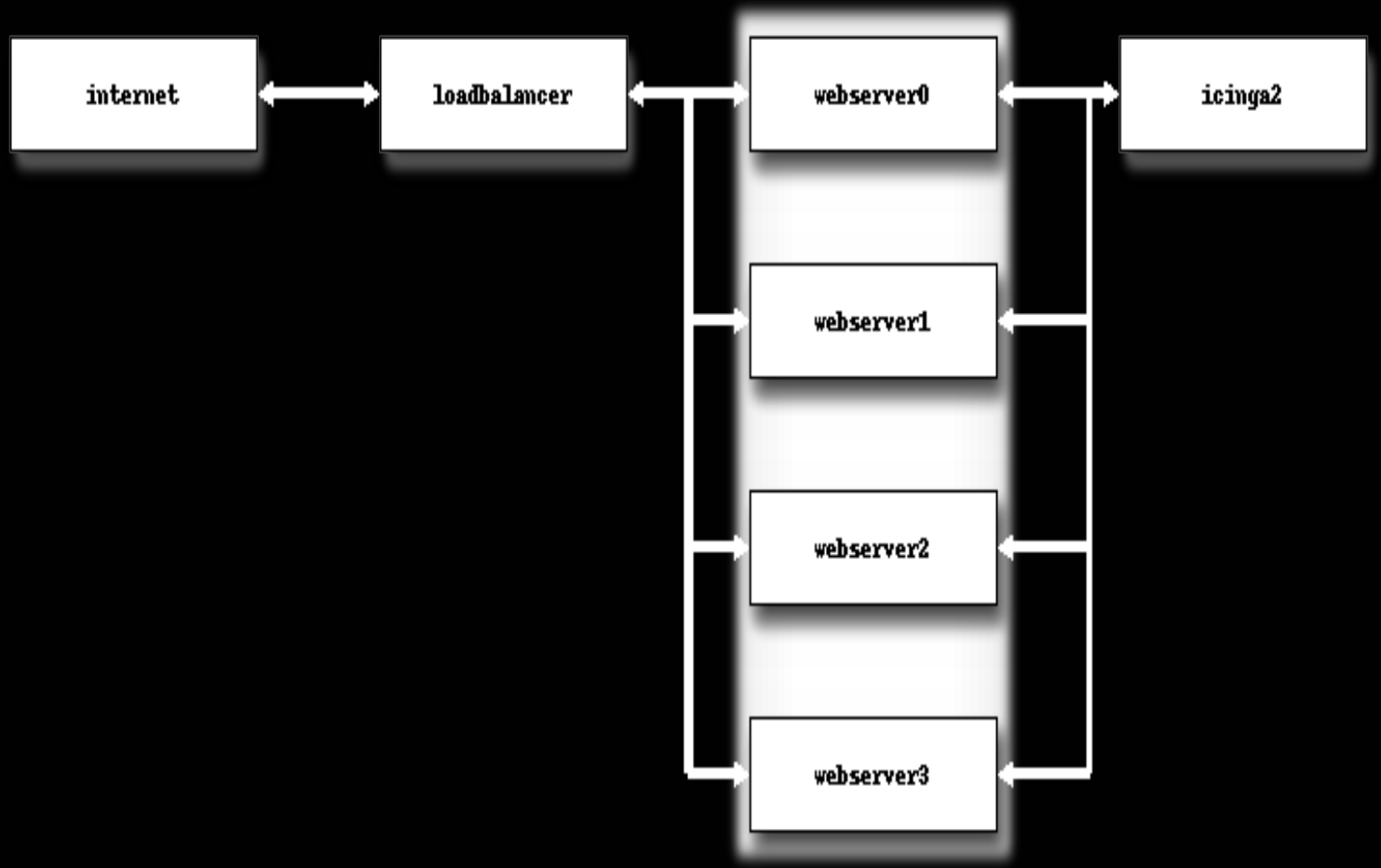
The End

Spoiler

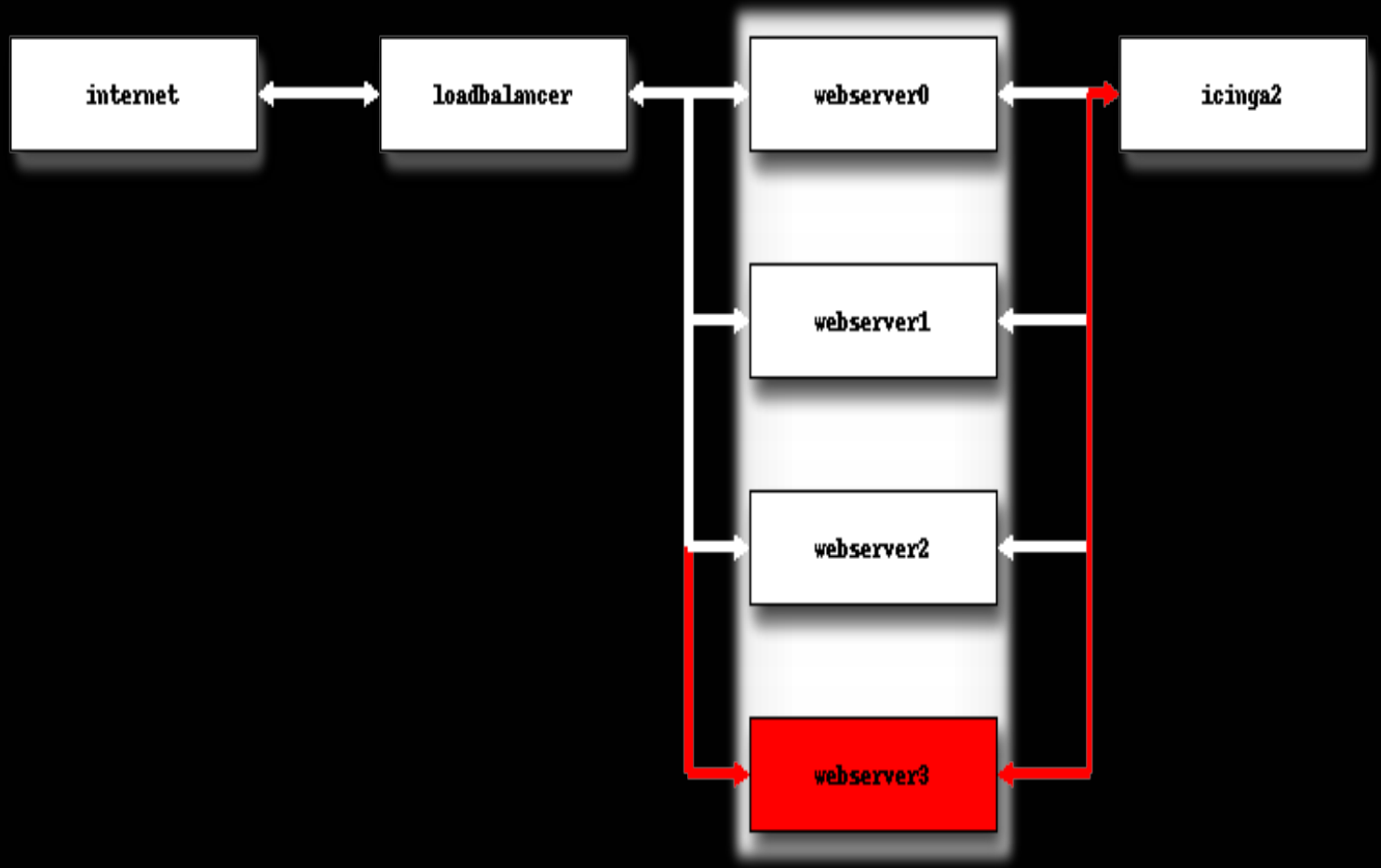
4

Examples

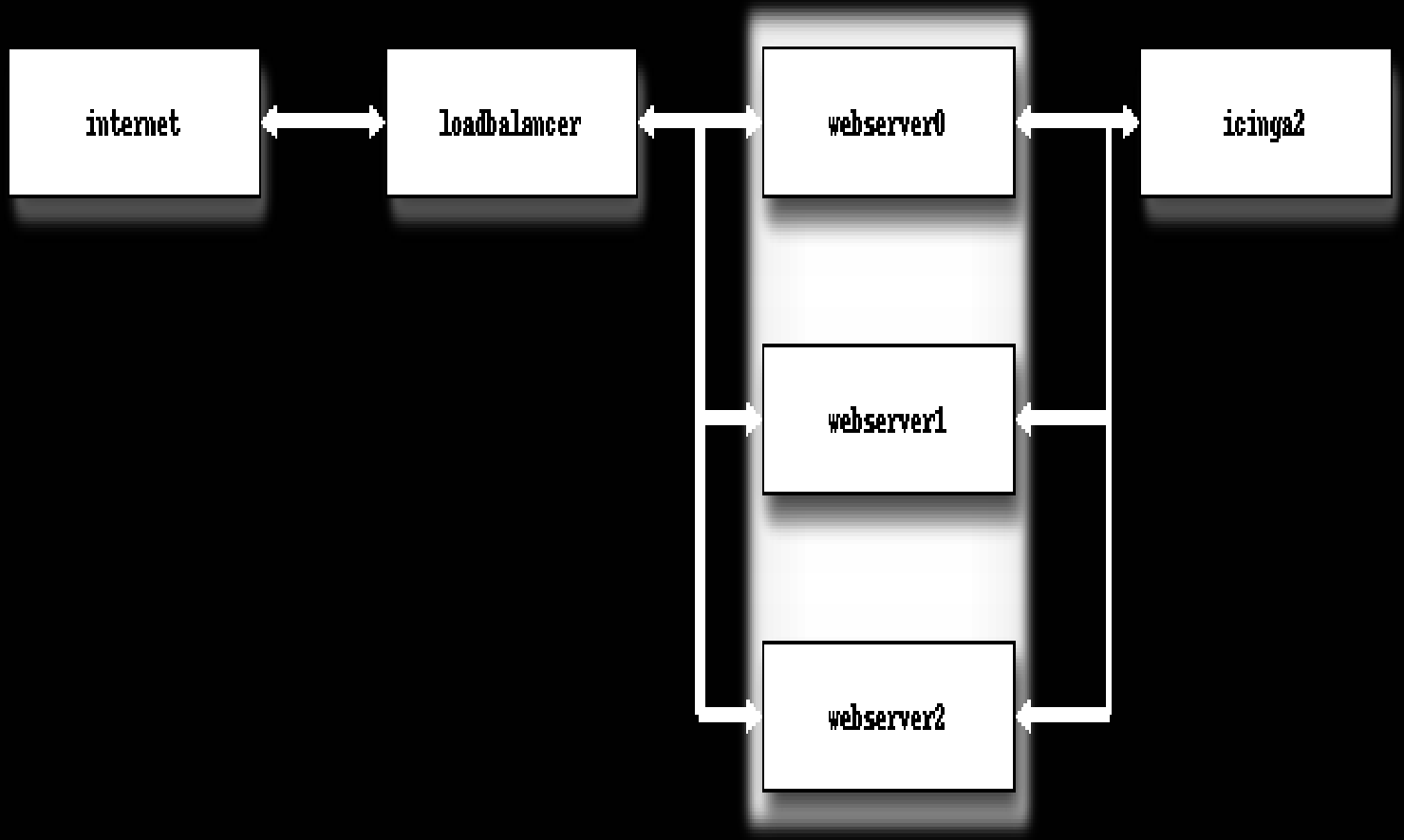
Load Balancer (0)



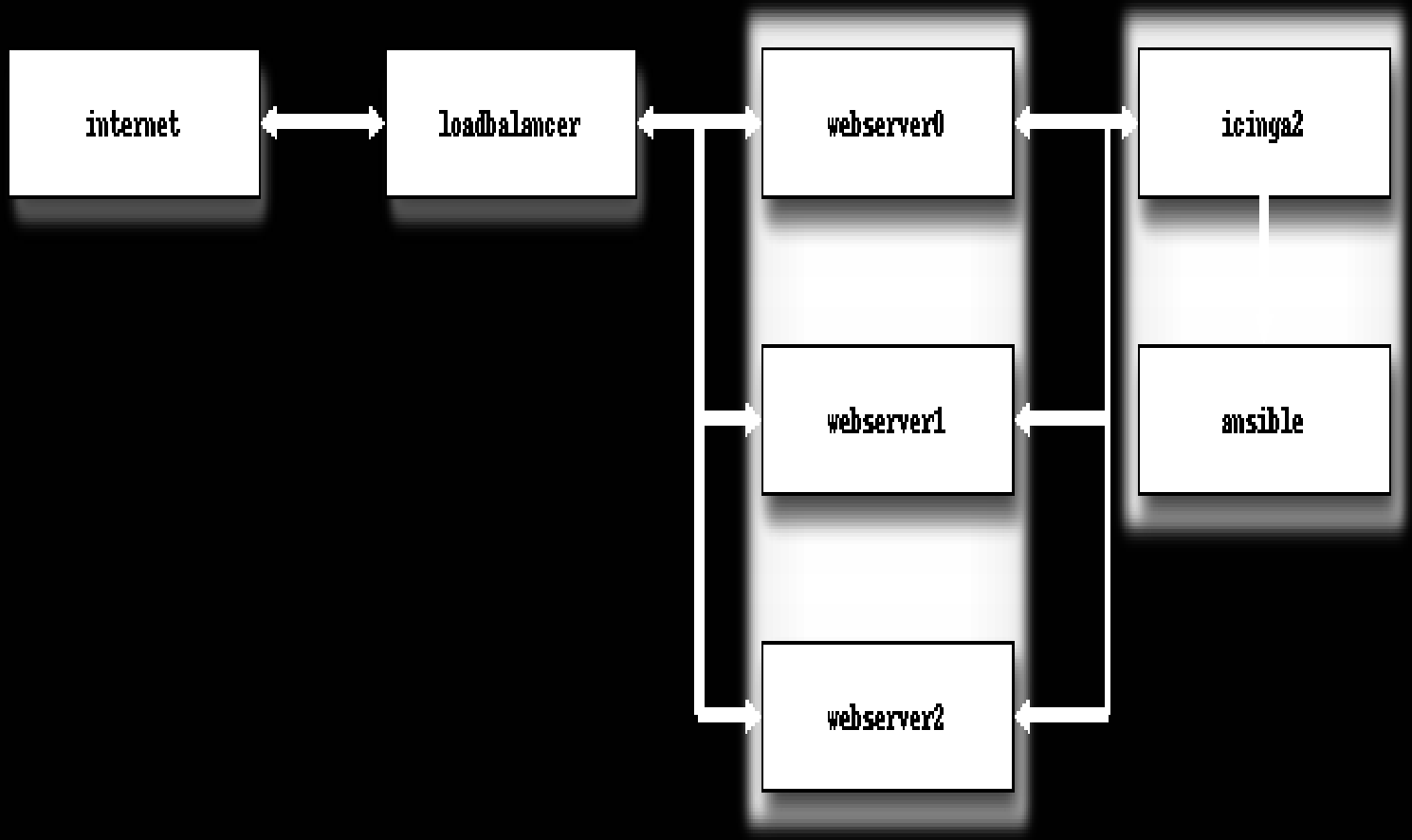
Load Balancer (1)



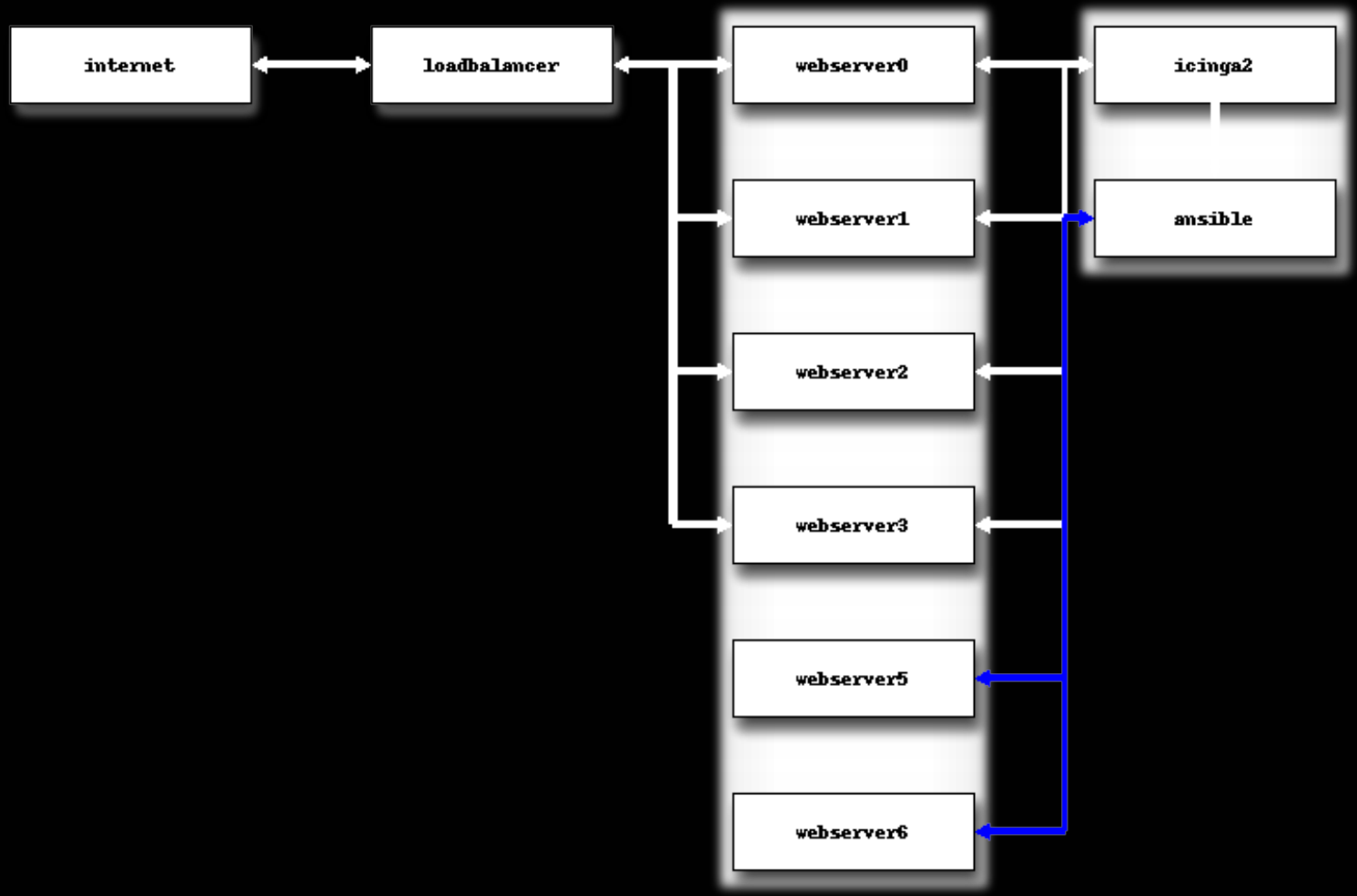
Load Balancer (2)



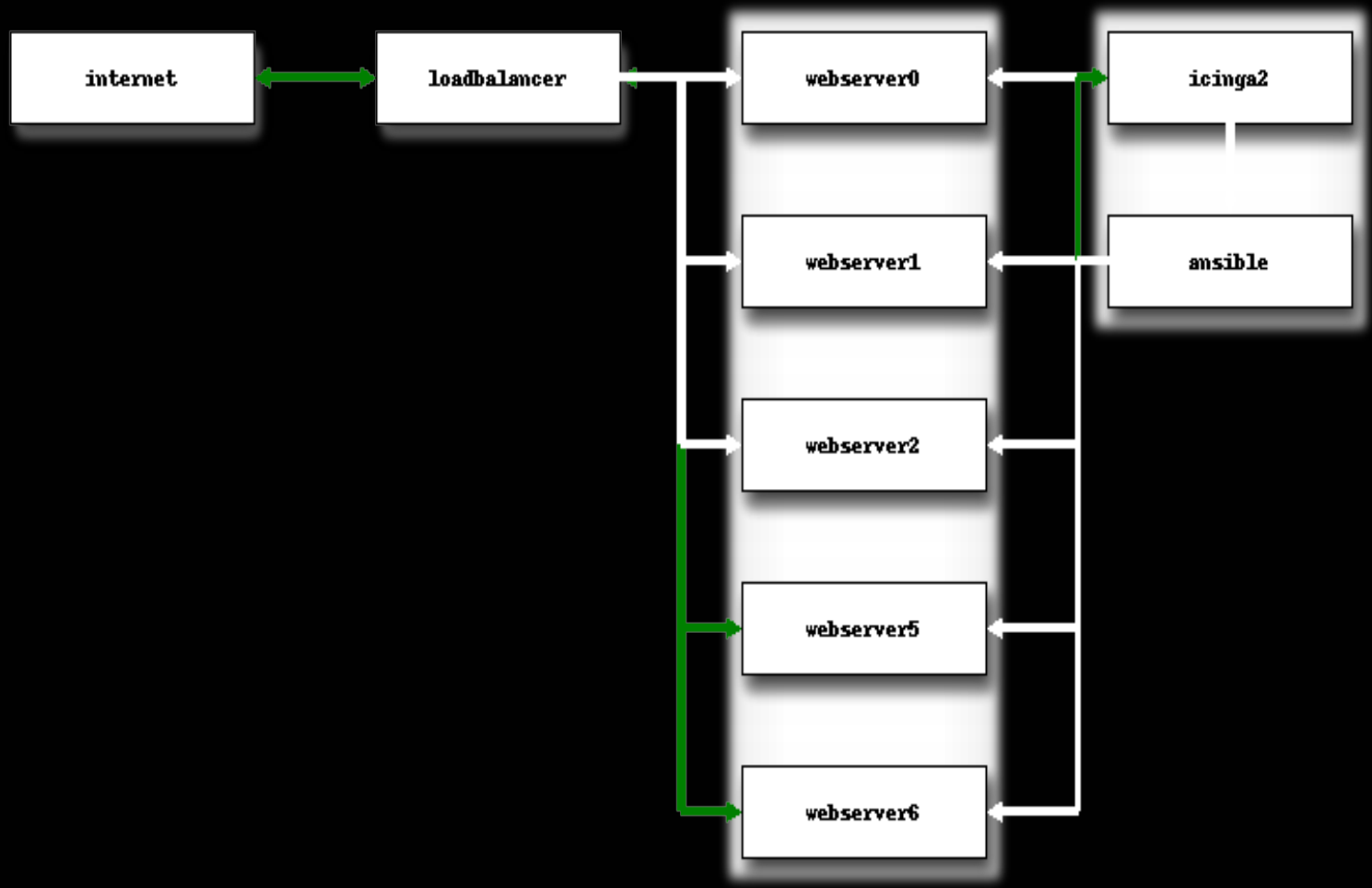
Load Balancer (3)



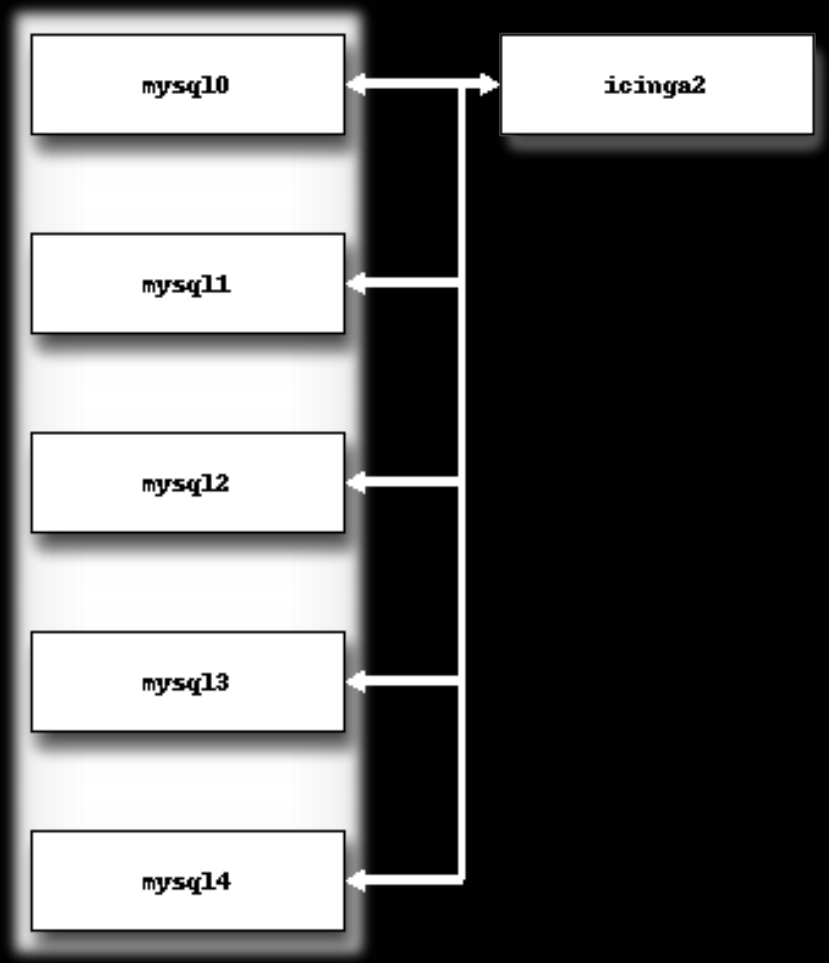
Load Balancer (4)



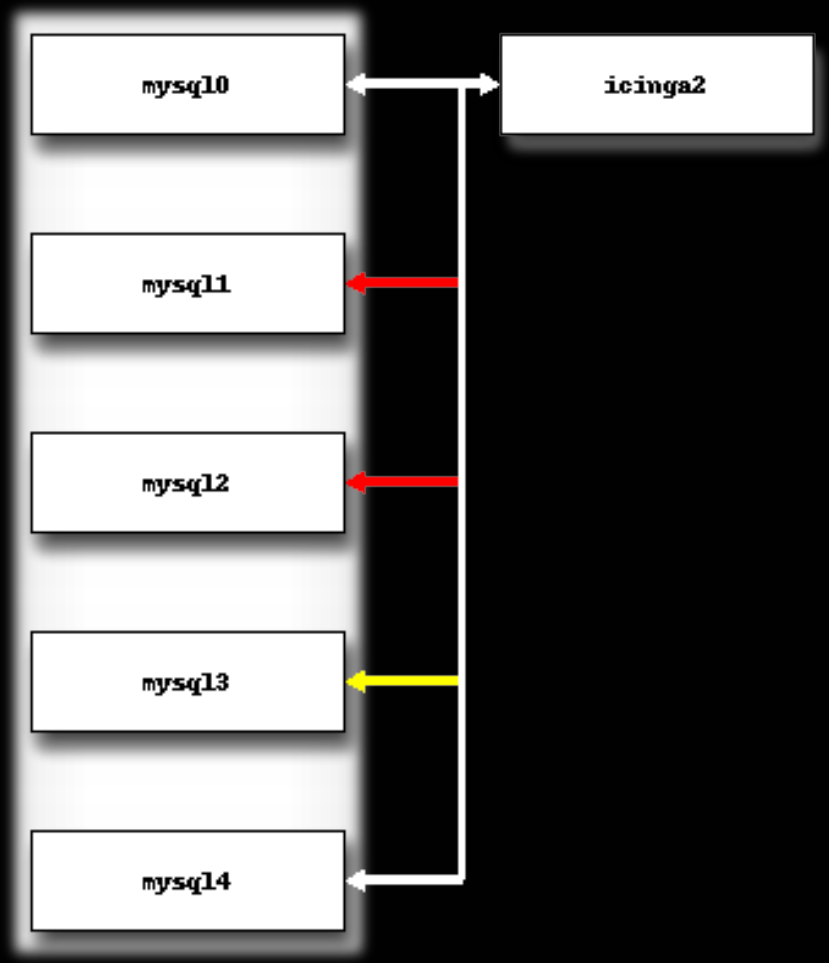
Load Balancer (5)



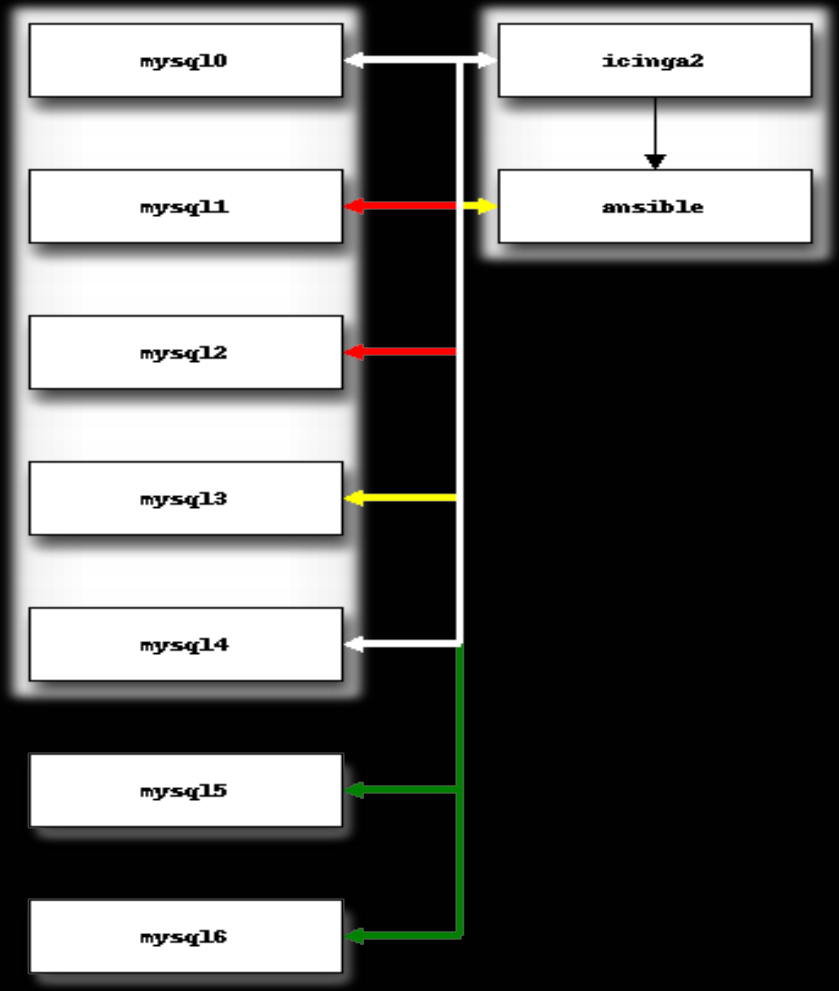
DB Cluster Repair (0)



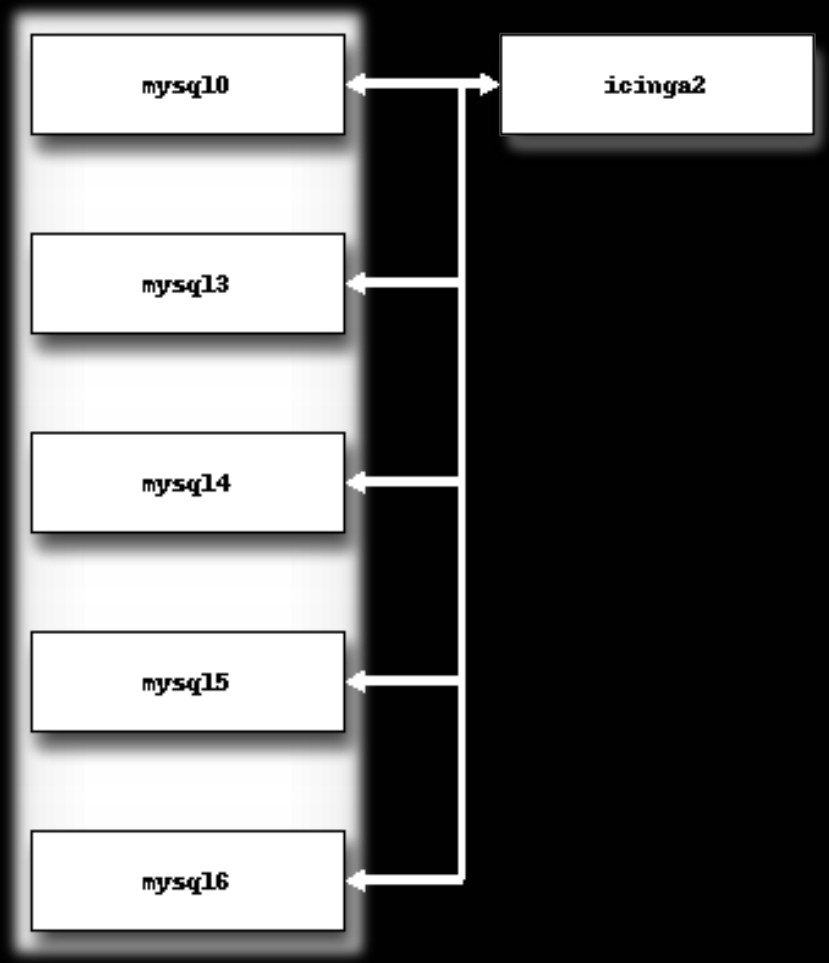
DB Cluster Repair (1)



DB Cluster Repair (2)



DB Cluster Repair (3)



The End

Thank You

Toshaan Bharvani - VanTosh bvba <toshaan@vantosh.com>

http://www.vantosh.com/

Made with Beamer LaTeX
a TeXbased Presentation program

CfgMgmtCamp 2020

3 $\rightarrow$ 5 February 2020

https://cfgmgmtcamp.eu/