# Smart Ring: A Model of Node Failure Detection in High Available Cloud Data Center

Lei Xu, Wenzhi Chen, Zonghui Wang, Huafei Ni, and Jiajie Wu

College of Computer Science and Technology, Zhejiang University,
Zheda Rd. 38, Hangzhou 310027, China
{leixu,chenwz,zjuzhwang,20921248,21121176}@zju.edu.cn

**Abstract.** Nowadays most of cloud data centers deploy high available system in order to provide continuous services, so it's very important for a high available cluster to detect the node failure (physical machine failure) accurately and timely in a low bandwidth occupation way. However, compared to the traditional cluster environment, the scale of cloud data center increases rapidly with the use of virtualization, so traditional node failure detection models have already faced several new problems. In this paper, we present a three roles and two layers node failure detection model, named as Smart Ring, which fits cloud data center well and strikes a balance between accuracy, instantaneity and bandwidth occupation. It can simultaneously detect the status of physical machines and virtual machines and deal well with multiple nodes failure and network partition. Our experiment results show that Smart Ring has a better performance than most existing models.

**Keywords:** Cloud Data Center, High Availability, Node Failure Detection.

## 1 Introduction

In the past few years, many kinds of cloud services spew out. These services must serve continuously for 24 hours a day and 365 days a year with minimal maintenance. In addition, in many important fields, such as Finance, Traffic, Telecom and Military, once the system crashes, even if a short stop running, it may bring unimaginable consequences. So cloud data center should deploy high available (HA) system to provide the most stable network service and check out the node failure in time.

In our research we have found several kinds of failure detection models that have been applied in HA system. However, these models designed for traditional clusters can't fit well with data center. Because the number of machines has exploded rapidly in data center, especially the numbers of virtual machines (VM), most existed models have obvious bottlenecks when they are applied in cloud data center. They either consume more time to detect node failure, or occupy more bandwidth or easy to make misjudgments. What's worse, some of these traditional models don't support virtualization architecture that means they can't monitor the status of VM efficiently.

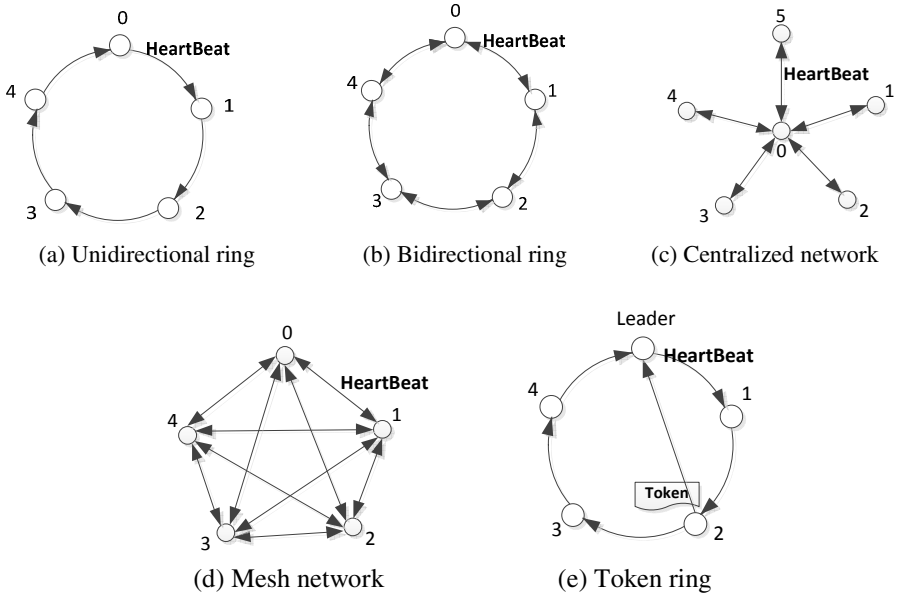
Considering the problems mentioned above, a novel model is proposed in this paper. We firstly introduce several classical node failure detection models in section 2.

J.J. Park et al. (Eds.): NPC 2012, LNCS 7513, pp. 279–288, 2012.

In section 3, we detail the design of Smart-Ring. Section 4 shows the extended functions of Smart-Ring. In section 5, we implement a prototype system to evaluate the performance of Smart Ring. A summary and plan of our future work are described in section 6.

## 2 Related Work

HA architecture has been extensively explored by cluster researchers in past years. This has led to the development of various node failure detection models. We have found many existing models used in HA system. These models can be summarized as five categories.




(a) Unidirectional ring (b) Bidirectional ring (c) Centralized network

(d) Mesh network (e) Token ring

**Fig. 1.** Classical node failure detection models

**Unidirectional ring** is the simplest model. G Rudolph [1] published a paper talked about parallel clustering on a unidirectional ring in 1993 and a patent of Symantec [2] is also based on unidirectional ring. This model is very easy to make a misjudgment, because its judge logic is too simple.

**Bidirectional ring** is an upgrade of the unidirectional ring. Wang [3] and Savari [4] et al. have discussed the bidirectional ring network model in their papers. This model can reduce misjudgment rate, because a node is judged invalid only when the precursor and successor both don't get heartbeat. However, it occupies double bandwidth as the unidirectional ring.

**Centralized network** is a kind of stellate reticulum model. The principle of Linux-HA project [5] is similar with centralized model, Khan et al. [6] and Hamlyn [7] have researched this centralized framework in distribution system. Nevertheless, if the

central one is down, the system can't keep running which means it is the bottleneck of system.

**Mesh network** is a kind of full connection model. Only when all other nodes can't get heartbeat from a node the system judges this node is definitely down. This model has the highest accuracy while occupies the highest bandwidth. A famous HA project named Linux Failsafe [8] developed by SGI & SUSE is based on this model.

**Token ring** is a classic network model. Nilausen [9], Hutchison [10] and Goapl [11] have done a lot of work around token ring model. This model can enhance reliability, so the grid fault-tolerant middleware GRM [12] and group communication project Corosync [13] both use token ring network model. How-ever, in this model, the cycle of fault detection is too long. It isn't suitable for applied in large scale virtual cluster yet.

Briefly, we can see that all these aforementioned models have their own features. But when focus on HA data center environment, they seem not to be suitable. Because they couldn't support virtualization architecture well and can't strike a balance among accuracy, instantaneity and bandwidth occupation. So based on long time of survey and study, we design a novel model which is fit for large scale HA data center.

## 3 A Novel Node Failure Detection Model - Smart Ring

Smart Ring has two monitoring layers as depicted in Fig. 2. At the physical machine (PM) layer, each node is monitored by precursor while VMs are monitored by their host PM at the virtual machine layer.

Smart Ring also has three node roles: leader, backup and common. When leader node is invalid, backup node will be elected to be new leader. When backup node is invalid, its first successor common node will change into backup. This process is irreversible. A short introduction about the features of these three roles should be made:

- Common node corresponds to an actual PM running with many VMs. Each common node has an IP list with three records corresponded to leader IP, backup IP and the first successor IP. With the first successor IP a common node can keep heartbeat with it, with leader IP a common node can inform a suspect failure to leader and backup IP will be the new leader IP after leader failure. This IP list is important for Smart Ring to maintain system operation.
- Leader node is also a common node when there are no failures, which means in most of time it works as a common server. This design can make Smart Ring have more equivalence. But the IP list stored in leader node is different with commons'. This IP list records backup IP and all common IP. Only when machines join, exit or malfunction, leader will do some special operations like detecting node failure, updating the IP list of related nodes or modifying the global view which is a relational map of all PMs & VMs.
- Backup node is also a common node just like the leader. The IP list stored in it records leader IP and all common IP. After leader modifies the global view, it should synchronize the view with leader through Sync Info Channel. And if leader is invalid, backup will become a new leader node. With the global view stored on it, it can take over leader's work.

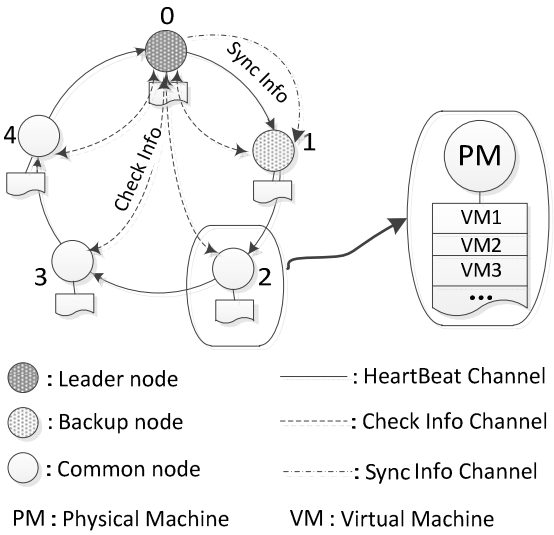



**Fig. 2.** Smart Ring model

### 3.1 Physical Machine Failure Detection

As mentioned above, nodes in Smart Ring have three different roles and the detection process of each role is different, we will describe respectively.

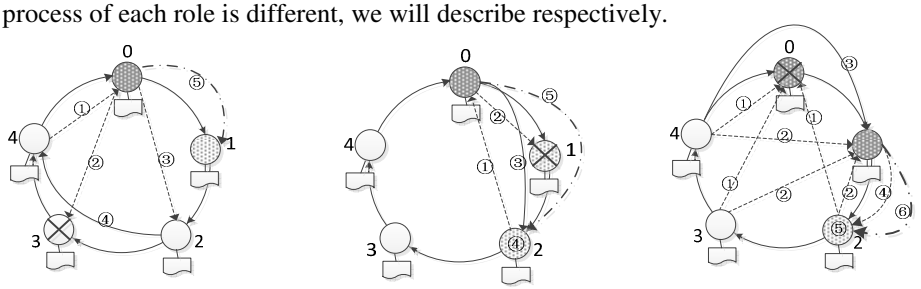



**Fig. 3.** Node Failure Detection Process. (a) Common node failure detection process. (b) Backup node failure detection process. (c) Leader node failure detection process.

**Common Node Failure Detection Process:** Now we assume node 3 is invalid, as shown in Fig. 3(a), node 4 wouldn't get heartbeat from 3 in appointed timeout. Then 4 will report this suspected fault to the leader node 0, 0 knows 4's precursor is 3 by checking the global view. 0 sends check info to 3 by Check Info Channel. If in the appointed timeout 0 can't get response from 3, in that way we think 3 is invalid, otherwise 3 is active. If 3 is indeed invalid, 0 will inform 4 to set its precursor as 3's precursor that is node 2, and 0 modifies global view and synchronizes with backup node 1. If node 3 is active, it indicates that there is only a link failure between 3 and 4, this link failure doesn't influence the normal work of node 3 and 4, so 0 will record

this link failure in system log file, 4 will monitor 3 again after system administrator fixes this link failure.

**Backup Node Failure Detection Process:** This process shown in Fig. 3(b) is similar with common node failure detection process. The only different is that if 1 is invalid, the leader node 0 will set node 2 which is the first successor of 1 as a new backup node and 0 sets its successor as 2. After that, 0 will update its successor info in its IP list, in the same way, 2 will update its precursor info too. Then 0 modifies global view and synchronizes it with the new backup node 2.

**Leader Node Failure Detection Process:** If leader is down, 1 wouldn't get the heart-beat from 0. Then 1 will inform all common nodes to check leader's status and then report the result to node 1. 1 will judge whether leader is active or not by all results. If most nodes (more than $Num_{All}/2$, $Num_{All}$ is the number of all nodes) judge leader is invalid, 1 will change itself into new leader and set its successor as new backup and then inform all nodes to modify their leader IP and backup IP info recorded in their own IP lists. Otherwise the leader is normal, and it is just a kind of link failure. Just like common node failure detection process we describe above, system writes down this failure in log file and 1 will monitor 0 again after administrator fixes this failure. This process is described in Fig. 3(c).

### 3.2 Virtual Machine Failure Detection

The process of virtual machine failure detection is relatively simple. As VMM (Virtual Machine Monitor) can get current status of virtual machines by commands, for example, use "xm list" command in Xen VMM. We should just set a suitable interval, and a daemon process executes this command periodically to get the status of virtual machines repeatedly. If host machine finds a VM fault, it will remove this VM info from the VM list stored in it and then report to leader. After leader gets this VM fault report, it immediately modifies global view and synchronizes with backup without checking again. The command we used is shown as follows:

```
xm list|sed -n '3,$'p|awk '{print $1;print $5}'> xmstatus.info
```

## 4 Extended Functions of Smart Ring

Node failure detection is just a major and primary function of Smart Ring. Be-yond that, it also has some extended functions which can deal with the problems in traditional HA system, such as multiple nodes fault and network partition.

### 4.1 Multiple Nodes Failure

In order to ensure that when multiple nodes fail simultaneously HA system still can work normally, we should just set more backup nodes. In a general way, if system meets the condition $Num_{Backup} \geqslant Num_{Fault}$ Where $Num_{Backup}$ is the number of backup nodes in supporting multiple nodes failure Smart Ring and $Num_{Fault}$ is the maximum

number of faulted machines that we want our system to support. This can ensure the normal work of system, because there are enough backup nodes could become leader when leader and several backup nodes fail simultaneously.

There is a new problem after Smart Ring sets multiple backup nodes that is how to keep global view consistency between leader and all backup nodes. For solving this problem, we can import a GCS (Group Communication System) in Smart ring. GCS is a private channel between leader node and all backup nodes, and it can ensure that all group members can receive messages orderly and reliably. Our research group has respectively deployed Apache ZooKeeper and Spread in our Smart Ring to keep global view consistency which are both excellent GCS toolkit.

### 4.2 Network Partition

Network partition [14] is the condition that exists after all network connections between any two groups of systems fail simultaneously. When this happens, systems on both sides of the partition can restart applications from the other side resulting in duplicate services, or split-brain. A split brain occurs when two independent systems configured in a cluster assume they have exclusive access to a given resource (usually a file system or volume). The most serious problem caused by a network partition is that it affects the data on shared disks. For solving this problem, Smart Ring takes such a strategy:

a) For a subgroup with leader node, it is a valid subgroup and keeps running only when it meets the condition $Num_{Now} \geq Num_{Most}/2$. Meanwhile, the other subgroups are invalid. $Num_{Now}$ is the number of this subgroup and $Num_{Most}$ is the number of nodes in the ring before network partition. $Num_{Most}$ is not a constant, its initial value is equal with the number of PMs and it will update after a network partition.

b) For a subgroup with backup node, first of all, backup changes into new leader. Then it counts up the number of nodes in this subgroup. If it meets the condition $Num_{Now} > Num_{Most}/2$, it is a valid subgroup and forms a new ring. So the other subgroups are invalid and they must quit Smart Ring.

c) For a subgroup with both leader and backup, it is treated as a valid subgroup no matter how many nodes in this subgroup. It will keep running while the other subgroups must quit HA system.

d) For a subgroup without both leader and backup, it is treated as an invalid subgroup no matter how many nodes in this subgroup.

Through this strategy, we can ensure that there is only an optimum sub-group valid and this subgroup can form a new Smart Ring to keep HA system continuously running.

## 5 Performance Evaluation

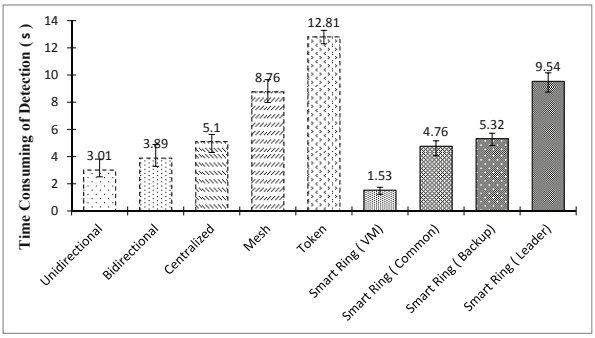
For evaluating the performance of Smart Ring, we have implemented a prototype system to observe the time consuming of node failure detection and the bandwidth occupation. Our experiments ran on this environment shown as Table 1:

**Table 1.** Experiment environment

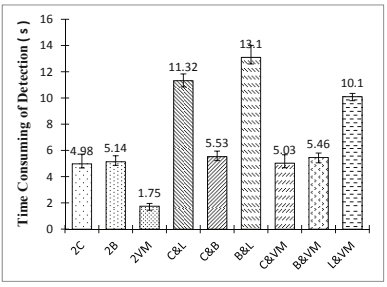
| PM Numbers | VM Numbers | CPU Frequency (GHz) | CPU Cores | Memory (GB) | Network (Mb/s) | Leader Numbers | Backup Numbers | Common Numbers |
|---|---|---|---|---|---|---|---|---|
| 64 | 6400 | 3.3 | 4 | 4 | 100 | 1 | 3 | 60 |

### 5.1 Node Failure Detection Evaluation

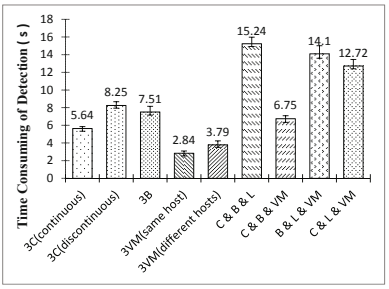
For the first experiment, we perform three groups of tests for 10 times each group. Firstly, we respectively calculate the average time consuming of one node failure detection among those five models we talked in section 2 and Smart Ring as a comparison. Then we calculate the average time consuming of two nodes failure detection in Smart Ring. At last, we calculate the average time consuming of three nodes failure detection which is the maximum number our experiment environment supports.




(a) One node failure detection (SmartRing VS other models)




(b) Two nodes failure detection




(c) Three nodes failure detection

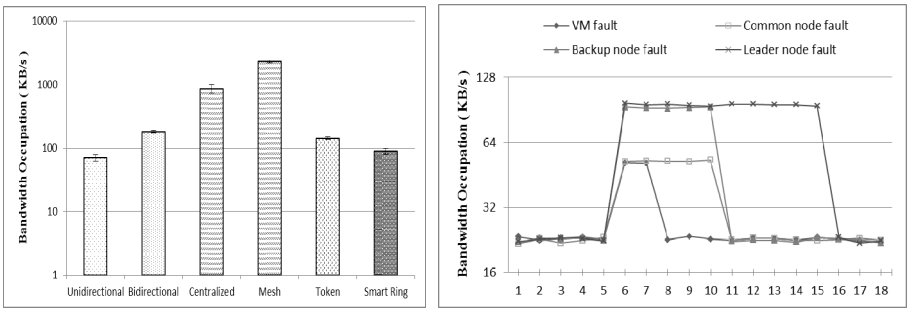
**Fig. 4.** Node Failure Detection Testing

As shown in Fig. 4(a), compared to other models, Smart Ring takes less time than the average level. Especially it spends the least time in VM failure detection which is the most possible failure in cloud data center. And we can see in the Smart Ring, VM

failure takes the least time, leader failure takes the most and common failure takes the same time as backup approximately. In Fig. 4(b), we examine the capacity of Smart Ring in two nodes failure detection where 2C means two Common nodes failure, 2B means two Backup nodes failure, C & L means a Common node and a Leader node failure and so on. From the results, we can see it takes the same time approximately as the situation of one node failure. We examine the capacity of three nodes failure detection in Fig. 4(c) as well. What to notice is that the time consuming of three continuous common nodes failure detection is different with those discontinuous and three VMs on the same host is different with those on the different hosts as well.

### 5.2 Bandwidth Occupation Evaluation

Smart Ring takes a tiny bandwidth occupation when HA system runs normally, because only heartbeat messages transferred between adjacent nodes occupy the bandwidth and VM failure detection executed inter physical machine doesn't occupy bandwidth neither. If there is a failure happens, bandwidth occupation will increase until a new ring forms. As we know the leader node which has the largest data flow is the performance bottleneck. So in our second testing, we compare the bandwidth occupation of leader node when a failure happens in Smart Ring with the other five models.




(a) SmartRing VS other models (b) Different node faults in SmartRing

**Fig. 5.** Bandwidth Occupation Evaluation

Fig. 5(a) shows the bandwidth occupation evaluation result among Smart Ring and other five models. Smart Ring is just a little higher than unidirectional ring model but a great fewer than others. From Fig. 5(b) we can know that Smart Ring just occupies about 22KB/s when HA system runs normally. But when a fault happens at the $5^{th}$ second it will increase until a new ring forms. For a VM fault or a common node fault the bandwidth occupation is about 55KB/s, while for a backup node fault or a leader node fault it is about 97KB/s. Because synchronizing the whole of global view occupies a little bandwidth. In addition, we can see the time from bandwidth occupation increases to it restores coincides with our first test results.

## 6 Conclusion and Future Work

In this paper, we have presented a three roles and two layers node failure detection model. To the best of our knowledge, this work is a kind of novel design and implementation of node failure detection that could be applied in a big scale cluster with plenty of VMs like a cloud data center. We have detailed how to detect a node failure quickly and exactly with a lower cost, support multiple nodes fail simultaneously, solve network partition problem. Moreover, we have evaluated Smart Ring within a 64*100 machines scale cluster. The result shows that Smart Ring has a higher accuracy compared to unidirectional and Bidirectional ring, a lower bandwidth occupation compared to centralized ring and mesh network, and a lower time consuming of node failure detection compared to token ring.

Nevertheless, our design has some limitations that we plan to address in the future. We now can't support discontinuous backup nodes and detect the status of VMs in a simple way through the VMM commands. Finally, it's just a beginning of Smart Ring of design, implementation and performance evaluation. Various other measurements and optimization strategies will need to be explored in the future.

## References

1. Rudolph, G.: Parallel clustering on a unidirectional ring. In: Proceedings of the 1993 World Transputer Congress on Transputer Applications and Systems, Aachen, Germany, September 20-22, vol. 36, p. 487. Ios Pr. Inc. (1993)
2. Pasqua, J.: Cluster communication in heartbeat messages. US Patent 7,330,444 (February 12, 2008)
3. Wang, L., Han, X.: Stability and hopf bifurcation analysis in bidirectional ring network model. Communications in Nonlinear Science and Numerical Simulation (2010)
4. Savari, S., Kramer, G.: The multimessage unicast capacity region for bidirectional ring networks. In: 2006 IEEE International Symposium on Information Theory, pp. 763–767. IEEE (2006)
5. Robertson, A.: Linux-ha heartbeat system design. In: Proceedings of the 4th Annual Linux Showcase and Conference, ALS 2000 (2000)
6. Khan, N., Mahajan, A.: Centralized framework with ring broadcasting for real time traffic in vehicular ad hoc networks. In: 2010 3rd International Conference on Emerging Trends in Engineering and Technology, CETET, pp. 842–847. IEEE (2010)
7. Hamlyn, A., Cheung, H., Yang, C.: Computer network distributed monitoring and centralized forecasting of utility distribution system operations. In: IEEE Canadian Conference on Electrical and Computer Engineering, CCECE 2008, pp. 001719–001722 (2008)
8. SGI: Linux failsafe (2009), `http://oss.sgi.com/projects/failsafe/`
9. Nilausen, J.: Token ring network management: Performance management. International Journal of Network Management 5(1), 47–53 (1995)
10. Hutchison, D., Coffield, D.: Simple token ring local area network. Microprocessors and Microsystems 8(4), 171–176 (1984)

11. Gopal, T., Raja, G., Vijaykumar, D., Sankaranarayanan, V.: Novel fault tolerant token ring network. Microelectronics Reliability 36(5), 707–710 (1996); Fault tolerance; Token ring networks
12. Siddesh, G., Srinivasa, K., Venugopal, K.: Grm: a reliable and fault tolerant data replication middleware for grid environment. In: Proceedings of the International Conference & Workshop on Emerging Trends in Technology, pp. 810–815. ACM (2011)
13. Dake, S., Caulfield, C., Beekhof, A.: The corosync cluster engine. In: Linux Symposium 85
14. Symantec: Network partition (1995), http://www.symantec.com/security_response/glossary/defin-e.jsp?letter=n&word=network-partition