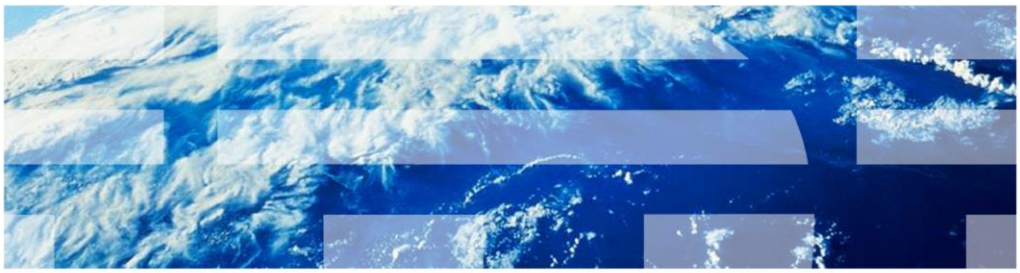
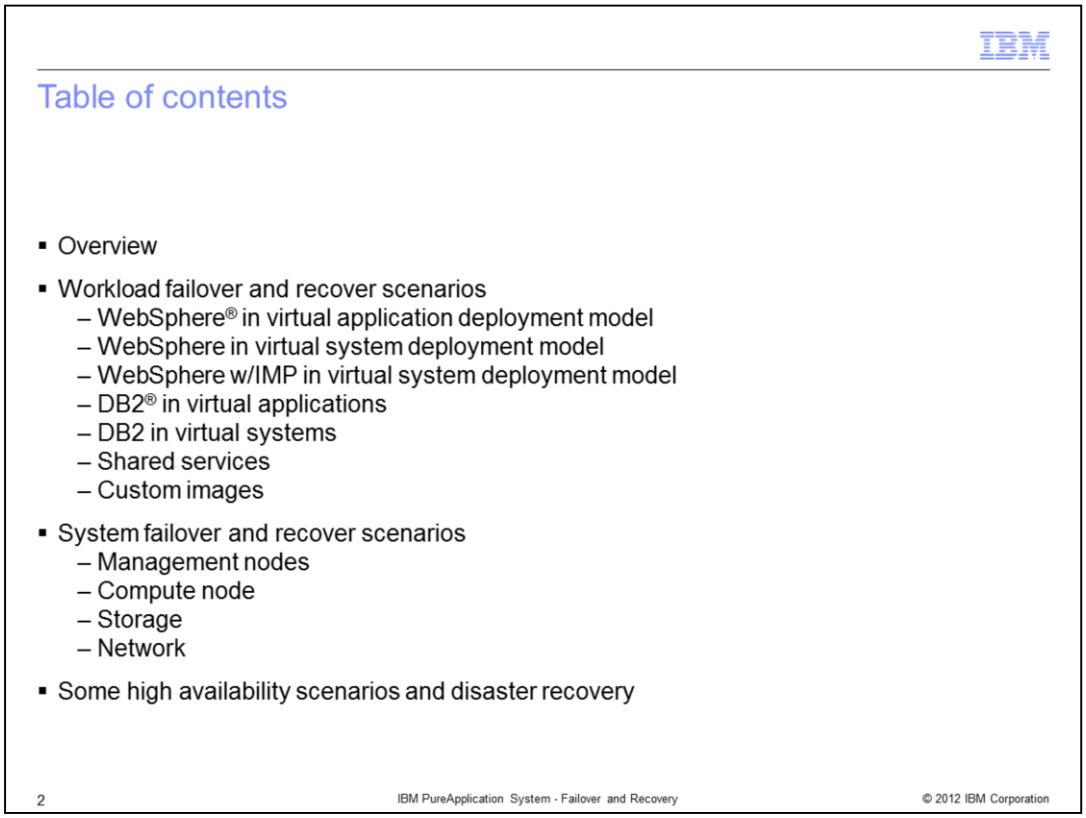
IBM PureApplication System

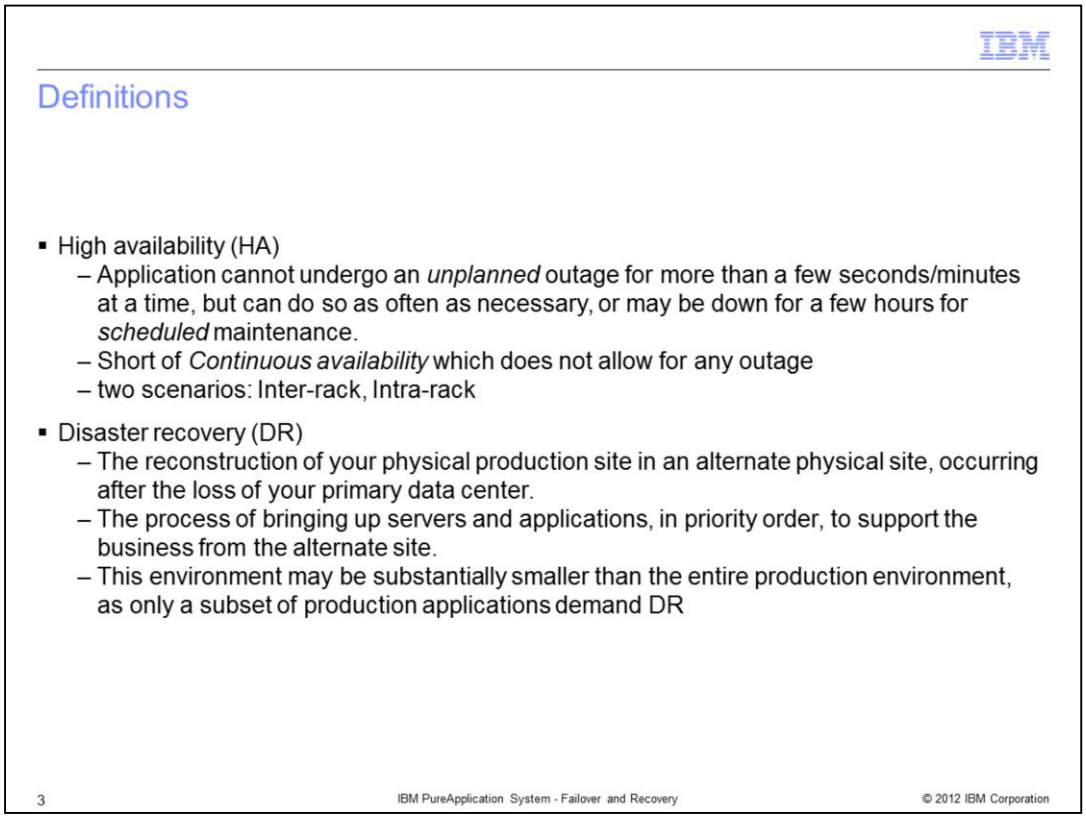
Failover and recovery scenarios



This presentation covers failover and recovery scenarios, both from workload and system perspective.



This presentation covers the workload and the system failover and recovery scenarios. Few examples of high availability are covered along with some discussion on disaster recovery.

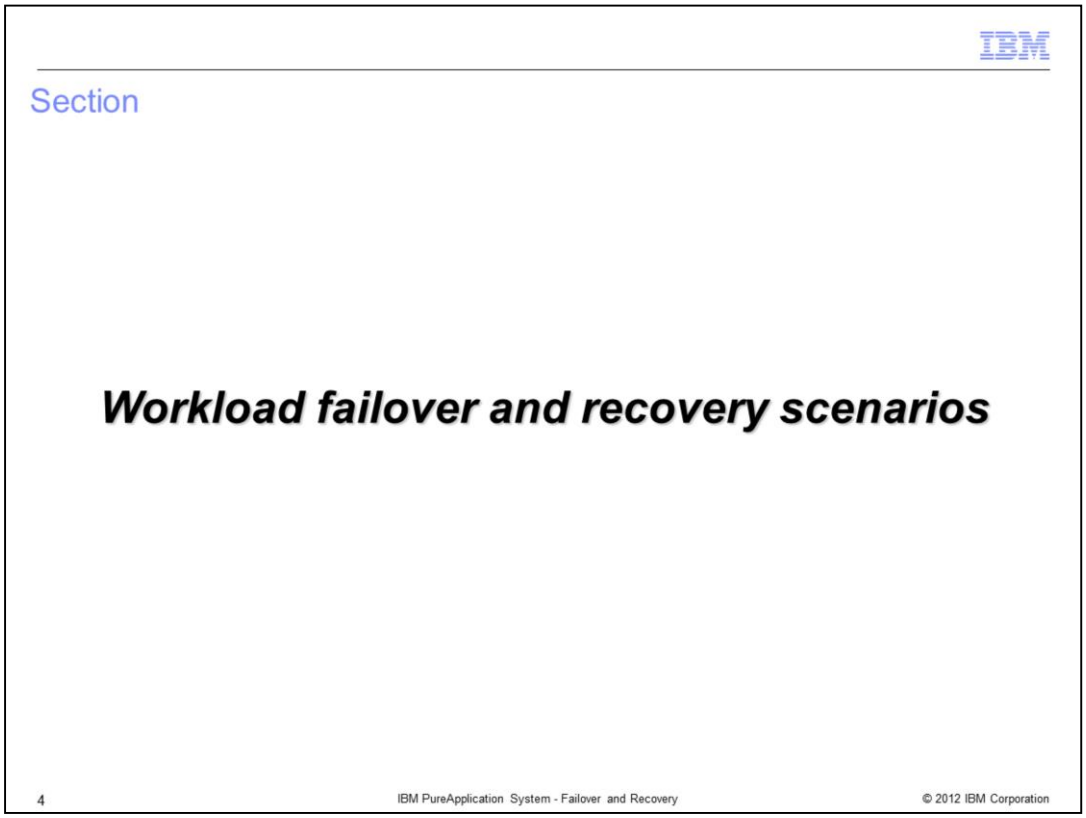


This slide covers some basic definition. High availability (HA) is required when applications cannot undergo an unplanned outage for more than a few seconds or minutes at a time, but can tolerate those short periods of not being available often, or can be down for a few hours for *scheduled* maintenance. This is short of what is defined as *Continuous availability* which does not allow or tolerate any outage.

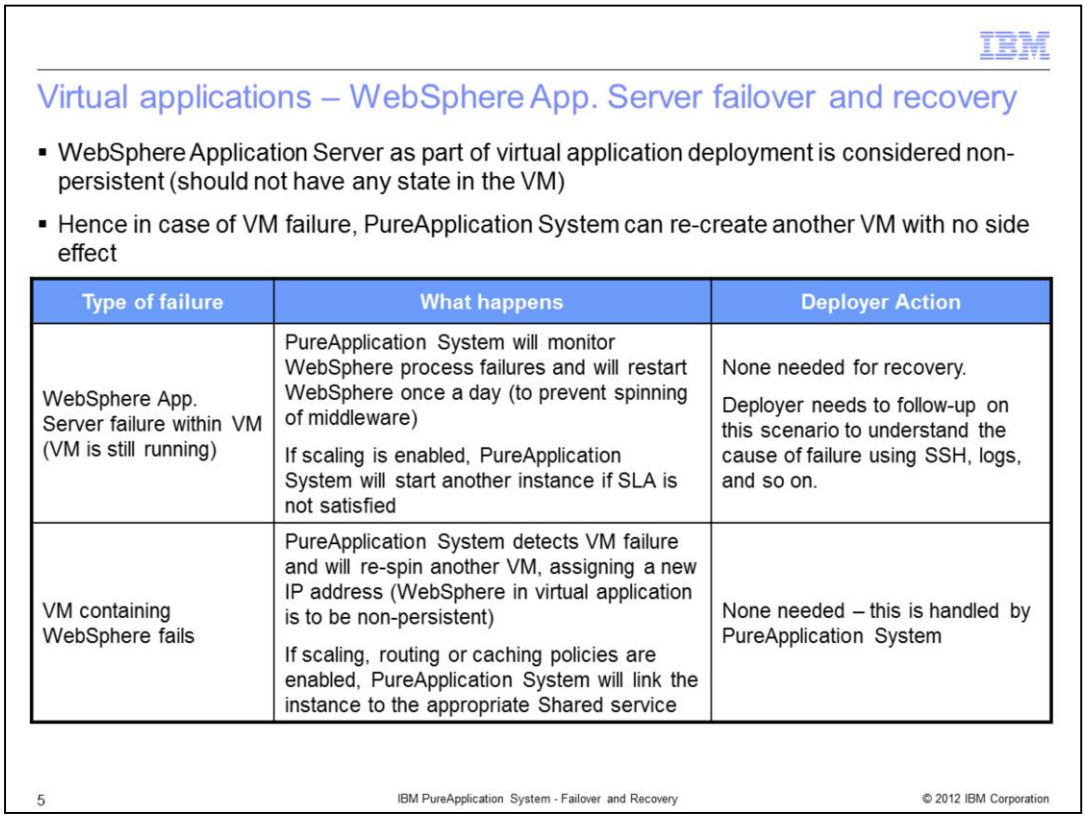
There are 2 scenarios to consider for HA. One is the HA features internal to the rack or intra-rack. The other is the HA features across rack or inter-rack. The next few slides will explore some of these scenarios

Disaster recovery is the ability to very quickly reconstruct and start the applications in an alternate physical site if the primary data center has some catastrophic loss and cannot continue to run the application for an extended period. It is the process of bringing the servers and applications in a priority order to support mission critical application at an alternate site. The alternate site need not be the same physical size as the primary data center since its role is to quickly get the mission critical application up and running.

This presentation covers the High availability scenarios and some recovery scenarios for consideration.



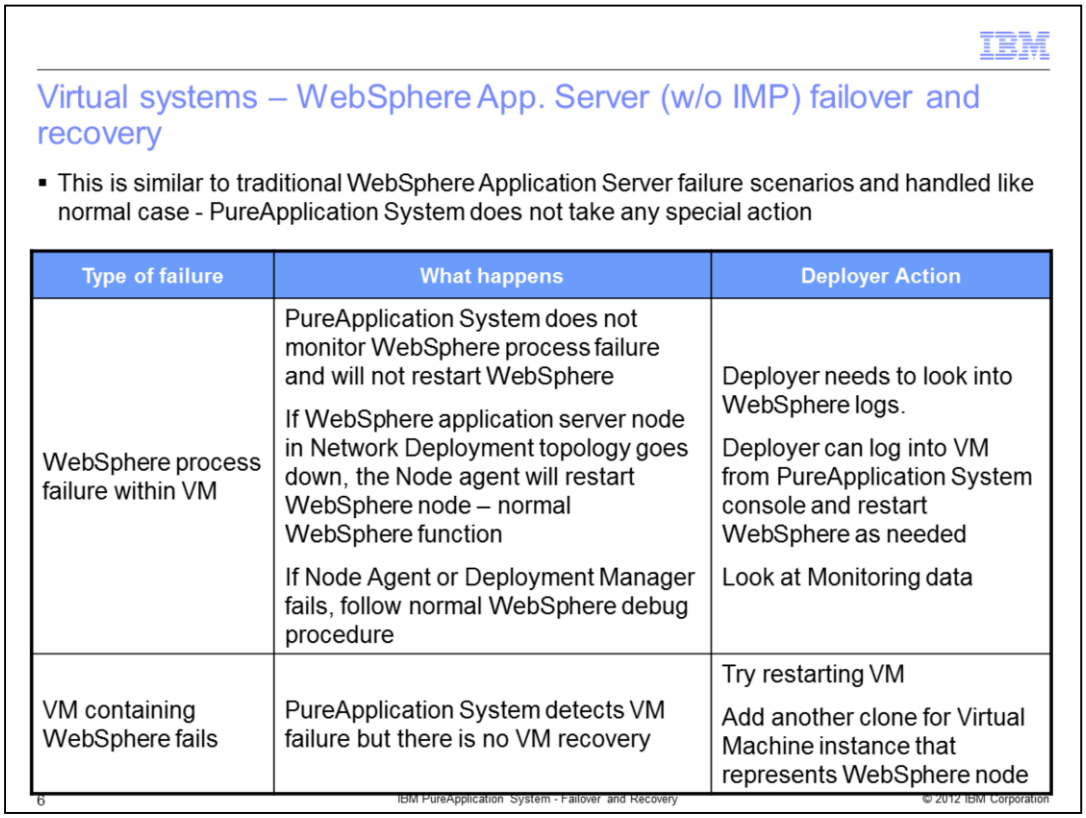
This section covers the workload failure and recovery scenarios for the different deployment models in PureApplication™ System.



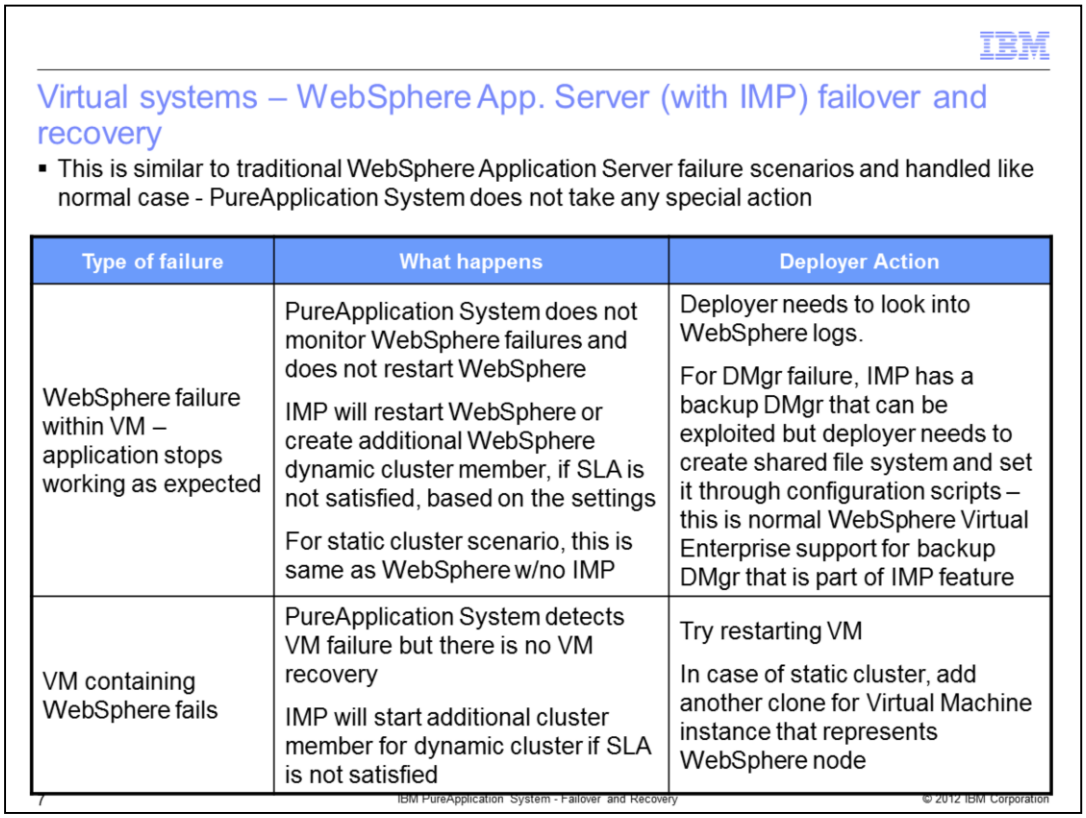
This slide discusses the failover and recover scenario of WebSphere Application Server running in virtual application deployment model. The basic assumption for WebSphere applications running in virtual application model is that the application is considered non-persistent. This means the application can run on any instance of WebSphere and there is no dependency that the state is locked in a specific instance. This means, in case of a failure of a WebSphere instance, PureApplication System can provision another WebSphere instance in another virtual machine.

If a WebSphere Application server fails within a virtual machine, and the VM is still running, PureApplication System will monitor WebSphere process failures and will restart WebSphere once a day to prevent spinning of middleware. If scaling is enabled, PureApplication System will start another instance if SLA is not satisfied. There is no action needed by the deployer. At some point, the deployer can determine the cause of the failure using the standard troubleshooting methods. However, if the VM itself fails, PureApplication System detects VM failure and will re-spin another VM, assigning a new IP address. If scaling, routing or caching policies are enabled, PureApplication System will link the instance to the appropriate Shared services. Again, there is no action required by the deployer. PureApplication System handles the recovery.

Best practice is to have scaling enabled so that PureApplication System can manage the recovery to satisfy the SLAs required by the application.



This slide discusses the failover and recover scenario of WebSphere Application Server without Intelligent Management Pack, running in virtual system deployment model. Virtual system WebSphere Application Server failure and recovery is handled by the built-in capabilities of the middleware and normal troubleshooting and debug practices need to be followed. PureApplication System does not monitor the WebSphere processes within the virtual machines in the virtual system deployment model. If WebSphere application server node in Network Deployment topology goes down, the Node agent will restart WebSphere node. This is normal WebSphere function. If Node Agent or Deployment Manager fails, you will have to follow normal WebSphere debug procedures to determine and fix the failure. The deployer can look at WebSphere log files, log into the VM and restart WebSphere, debug from within the VM or look at the monitoring data for potential cause of failures. If the VM running WebSphere fails, PureApplication System detects the VM failure but there is no VM recovery. Deployer can try to restart the VM from the PureApplication System workload console, add additional nodes by cloning the deployed instance, while they debug the root cause of the failure.



This slide discusses the failover and recover scenario of WebSphere Application Server with Intelligent Management Pack, running in virtual system deployment model. Intelligent Management pack provides dynamic clustering capability from the WebSphere Virtual Enterprise product. This capability allows WebSphere to provision additional servers to satisfy SLAs, or when excess capacity is present, remove some servers and still maintain the SLAs when the load goes down.

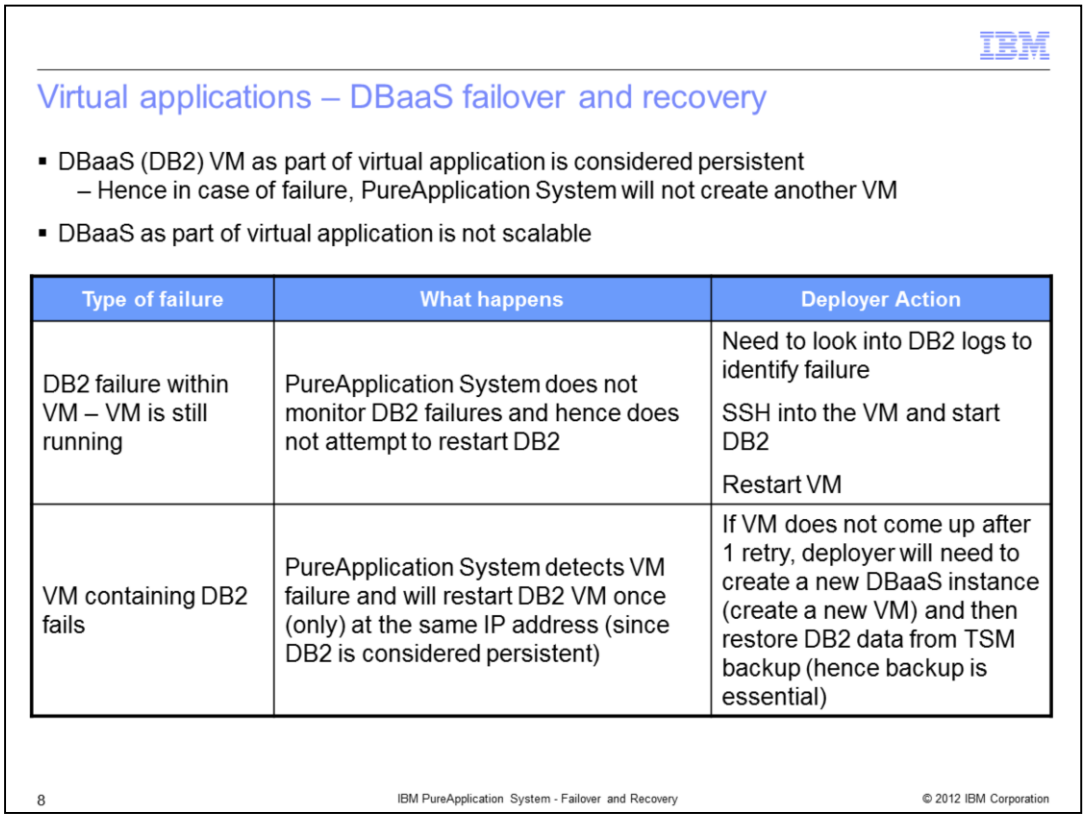
Just like the last use case, virtual system WebSphere Application Server with IMP failure and recovery is handled by the built-in capabilities of the middleware and normal troubleshooting and debug practices need to be followed. PureApplication System does not monitor the WebSphere processes within the virtual machines in the virtual system deployment model.

IMP will restart WebSphere or create additional WebSphere dynamic cluster member, if SLA is not satisfied, based on the settings.

If static cluster is used in WebSphere with IMP enabled, this use case is the same as WebSphere with no IMP as described in the last slide.

For deployment manager failure, IMP has the capability to support backup deployment manager, as long as the shared file system is configured, as explained in the WebSphere Virtual Enterprise information center.

If the VM running WebSphere fails, PureApplication System detects the VM failure but there is no VM recovery. For dynamic cluster, if the SLA is not satisfied, IMP will start additional cluster member. Deployer can try to restart the VM from the PureApplication System workload console, add additional nodes for static clusters by cloning the deployed instance, while they debug the root cause of the failure.

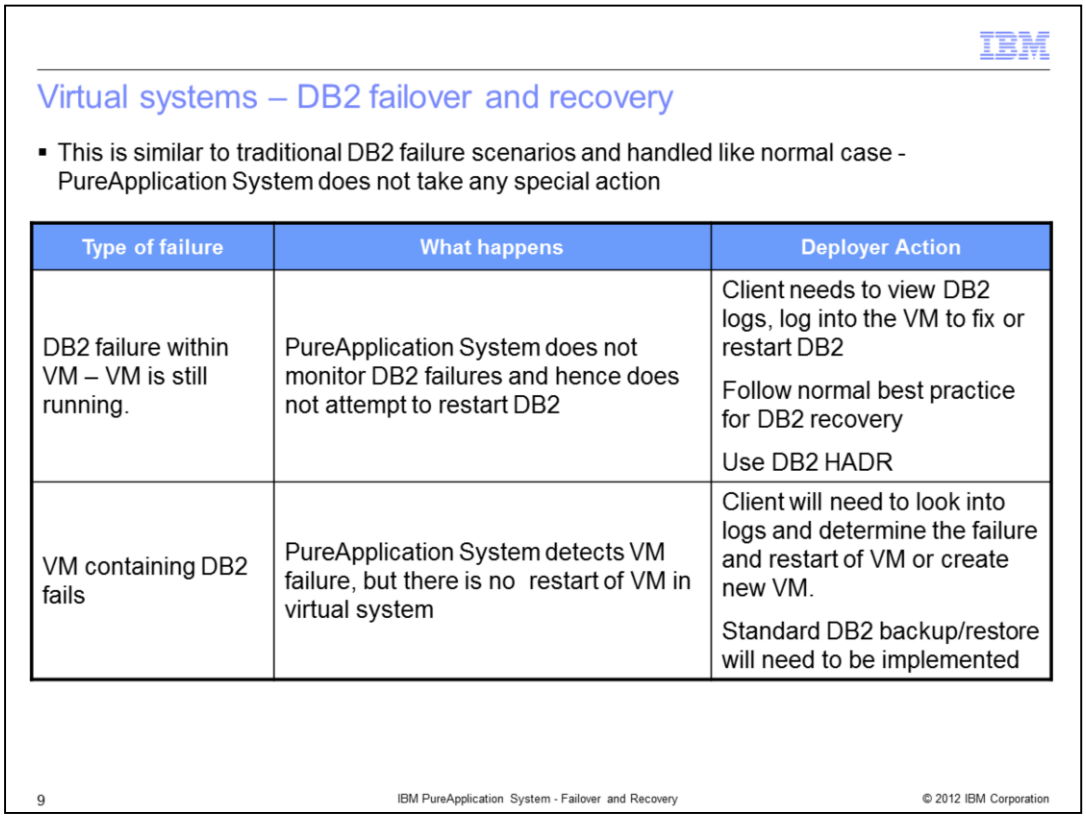


This slide discusses the failover and recover scenario of DB2 running in virtual application deployment model.

The virtual machine for Database-as-a-Service or DBaaS as part of DB2 virtual application deployment is considered persistent. It has data and hence a state. This means, in case of failure, PureApplication System cannot just spin another database instance. Additionally, in the current release, DBaaS as part of virtual application is not scalable.

PureApplication System does not monitor DB2 failures. Hence, if there is DB2 failure and the VM is still running, PureApplication System does not attempt to restart DB2. Deployer will need to look into DB2 logs to identify failure by getting in the VM by way of SSH and if needed, restarting DB2.

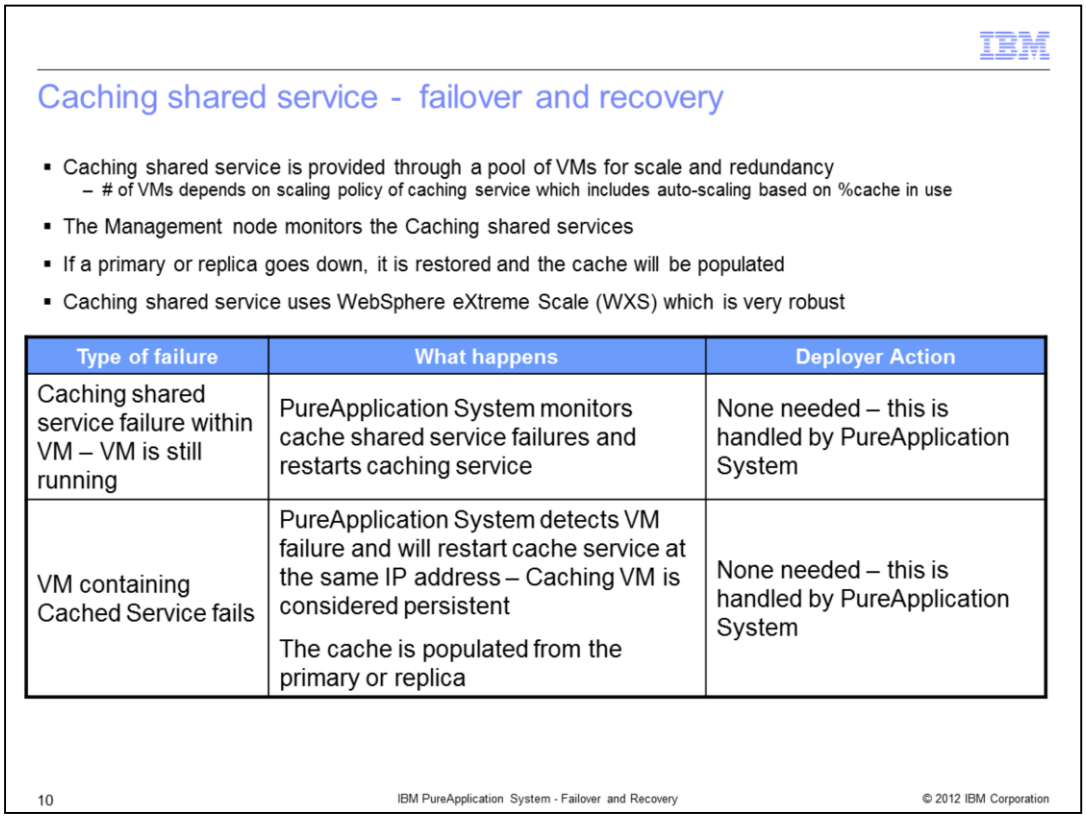
If the VM containing DB2 fails, PureApplication System detects VM failure and will restart DB2 VM once at the same IP address, since DB2 is considered persistent. If VM does not come up after 1 retry, deployer will need to create a new DBaaS instance VM and then restore DB2 data from Tivoli® Storage Manager backup. Hence for use cases where data loss cannot be tolerated, periodic backups are essential. Deploy SSH into the VM and start DB2; restart the VM.



This slide discusses the failover and recover scenario of DB2 running in virtual system deployment model. This is similar to traditional DB2 failure scenarios and handled like normal DB2 failures. Normal DB2 troubleshooting techniques apply. PureApplication System does not take any special action.

If the DB2 failed but the VM is still running, PureApplication System does not monitor DB2 failures and hence does not attempt to restart DB2. Clients will need to view the DB2 logs, log into the DB2 VM to fix or restart DB2. Normal DB2 best practice and recovery process applies to DB2 running in PureApplication System. PureApplication System contains DB2 Enterprise, and DB2 HADR images. For clients needing HADR capability can use the primary and secondary DB2 HADR virtual image parts.

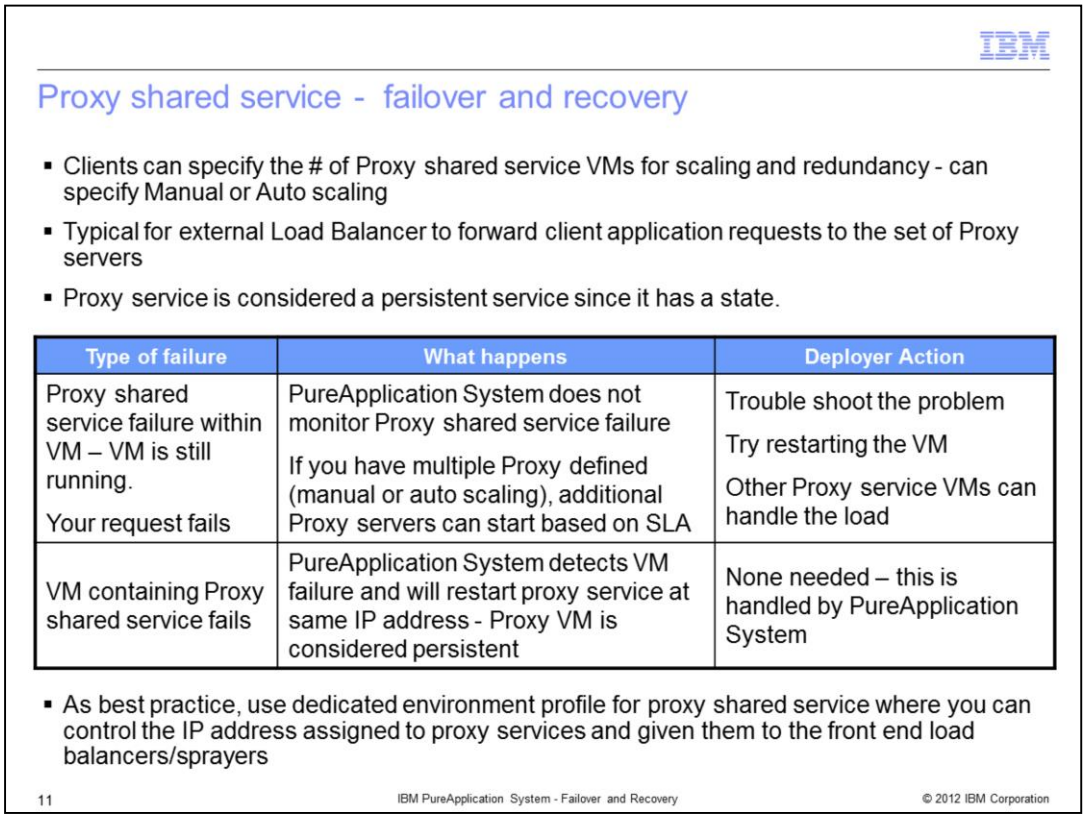
If the VM containing DB2 instance failed, PureApplication System detects VM failure, but there is no restart of VM in virtual system. The client will need to look into logs and determine the failure and restart the VM or create a new VM. Standard DB2 backup/restore will need to be implemented. DB2 HADR can help here where the secondary DB2 can take over processing the requests.



This slide discusses the failover of caching services used by WebSphere applications running as part of virtual applications.

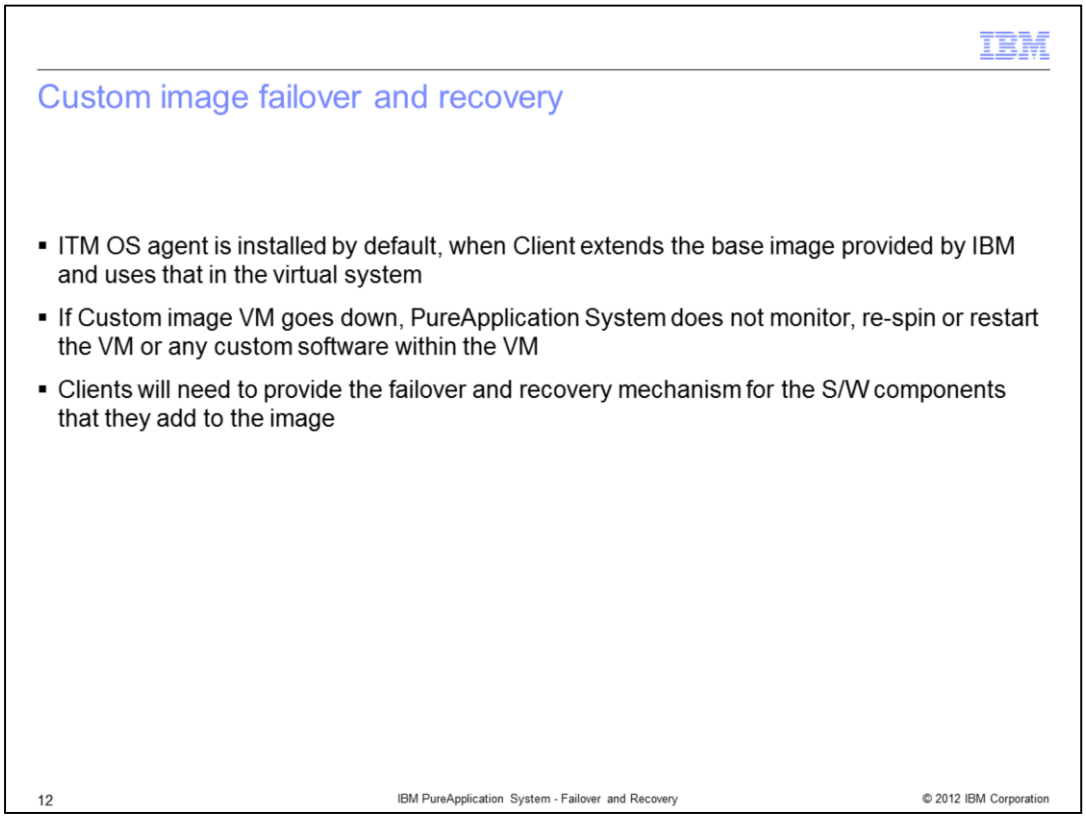
Caching shared service is provided through a pool of VMs for scale and redundancy. The number of VMs depends on scaling policy of caching service which includes auto-scaling based on the percentage cache in use. Caching shared service uses WebSphere eXtreme Scale (WXS) which is very robust. Internally, Extreme Scale has its own monitors that monitors the caching shared services.

If a caching shared service fails within a VM and the VM is still running, PureApplication System restarts caching service. There is no action needed by the deployer. Caching VM is considered persistent VM. Hence if a VM fails, PureApplication System detects VM failure and will restart cache service at the same IP address. There is no action needed by the deployer. If a primary or replica goes down, it is restored and the cache will be populated. PureApplication System handles the failures.



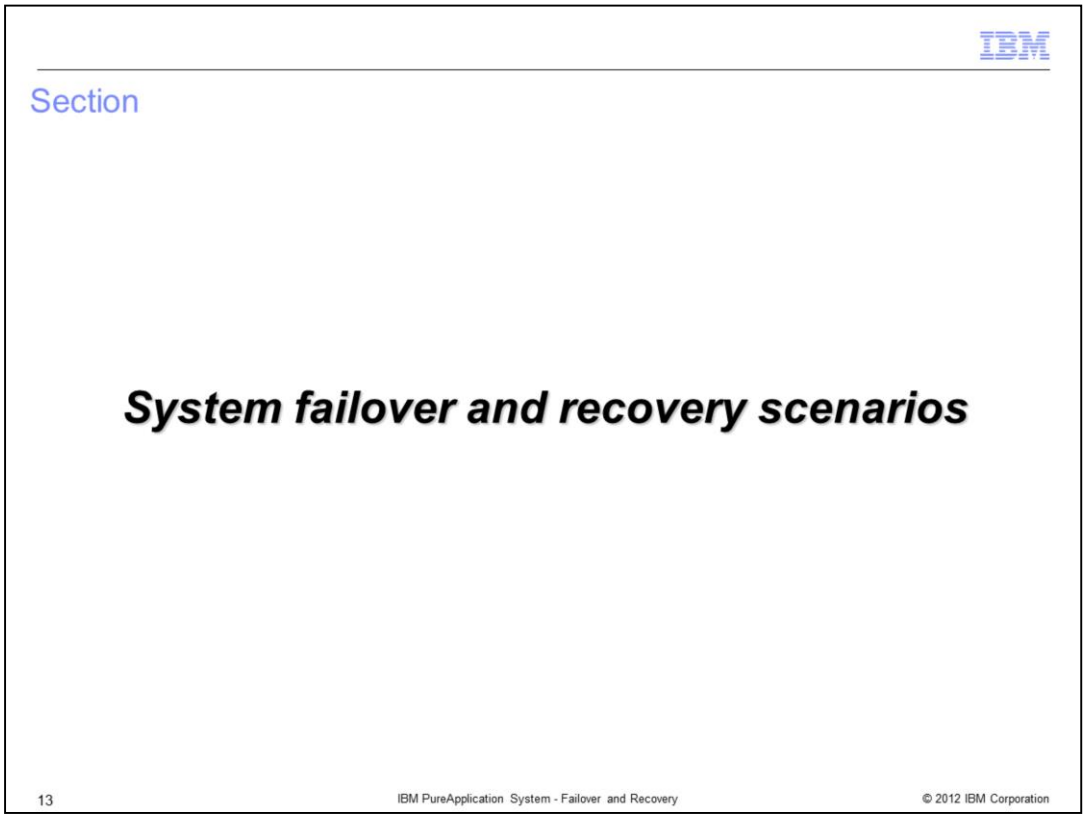
Proxy service supports scaling. Deployer can specify manual or auto scaling. These set of proxy servers are typically front ended by external load balancer. PureApplication System does not monitor Proxy shared service failure. If the proxy shared service instance fails but the VM is still running, PureApplication System does not restart the proxy service. However, if you have multiple proxy defined either through auto scaling, additional proxy servers can start based on SLA. In the meantime, you can troubleshoot the problem and try to restart the VM. Since the proxy service is considered persistent, if the VM containing the proxy service fails, PureApplication System detects VM failure and will restart proxy service at the same IP address.

As a best practice, you can either use fixed IP address for the proxy services VM or have a separate smaller pool of IP group for proxy service. These set of IP addresses can then be given to the front end load balancer to spray the requests. One way to achieve this is to have a separate environment profile just for proxy service and then use that for deployment of proxy shared services.



This slide covers the custom image failover and recovery. Custom image can be created using IBM Composition and Construction tool or using the PureApplication System extend and capture function. Details of tool and the extend/capture function is discussed in a separate presentation.

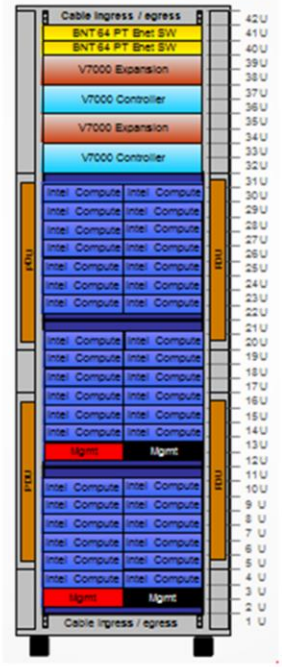
If you use one of the IBM supplied images for base virtual image as the base image for creating a custom image, it includes ITM OS agent to monitor the base OS. All failover and recovery functions will have to be handled by you in the image. PureApplication System does not monitor, re-spin or re-start the VM or any custom software within the VM.



This section covers the system hardware failover and recovery scenarios.

Hardware high availability

- **Management nodes and Virtualization mgmt node**
  - Both have redundant backup servers
  - Floating IP address is assigned to the active Management Node that has management (workload deployer) function
- **Network controllers**
  - Switches and cabling are redundant (2 of each).
  - Failure of 1 leads to reduced bandwidth, not service
- **Storage controllers**
  - Each controller has two canisters that can service all traffic needed
  - If one fails, the other handles all I/O
- **Storage**
  - SSD and HDD storage is configured in RAID5 + spares
    - Tolerates 2 concurrent drive failures without data loss (after spares are in use)
- **Compute nodes**
  - Management system will route around failed DIMMs or cores
  - If entire node fails, PureApplication System will try to move the VM to another compute node within its Cloud Group if free space is available – this is called "Rebalance or Workload evacuation"
    - If limited resources available on other nodes to handle the extra load, VMs are started based on their priorities

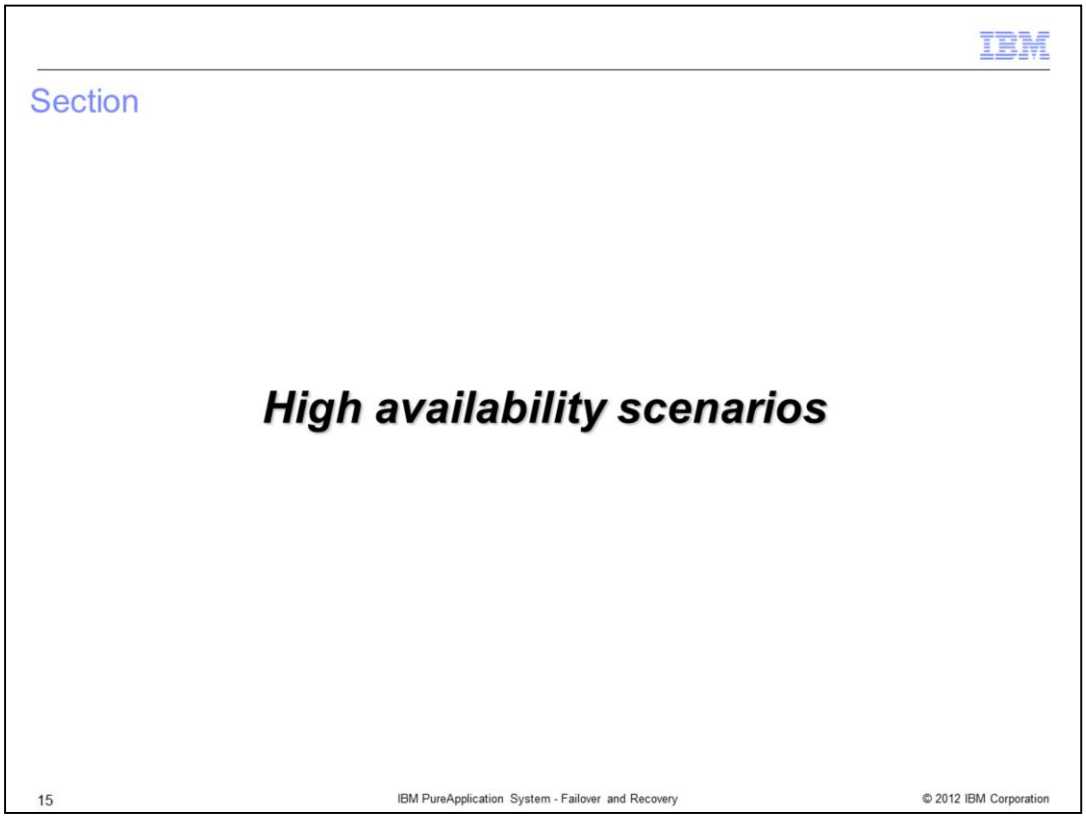


PureApplication System has built-in hardware redundancies for failover. Both the management node and the virtualization node have redundant backup servers that are continuously kept in sync with each other. PureApplication System has a floating IP address that is used to access the PureApplication System management functions. If one of the management nodes goes down, the floating IP address is automatically assigned to the backup server and thereby requiring no change by the clients to access the management function.

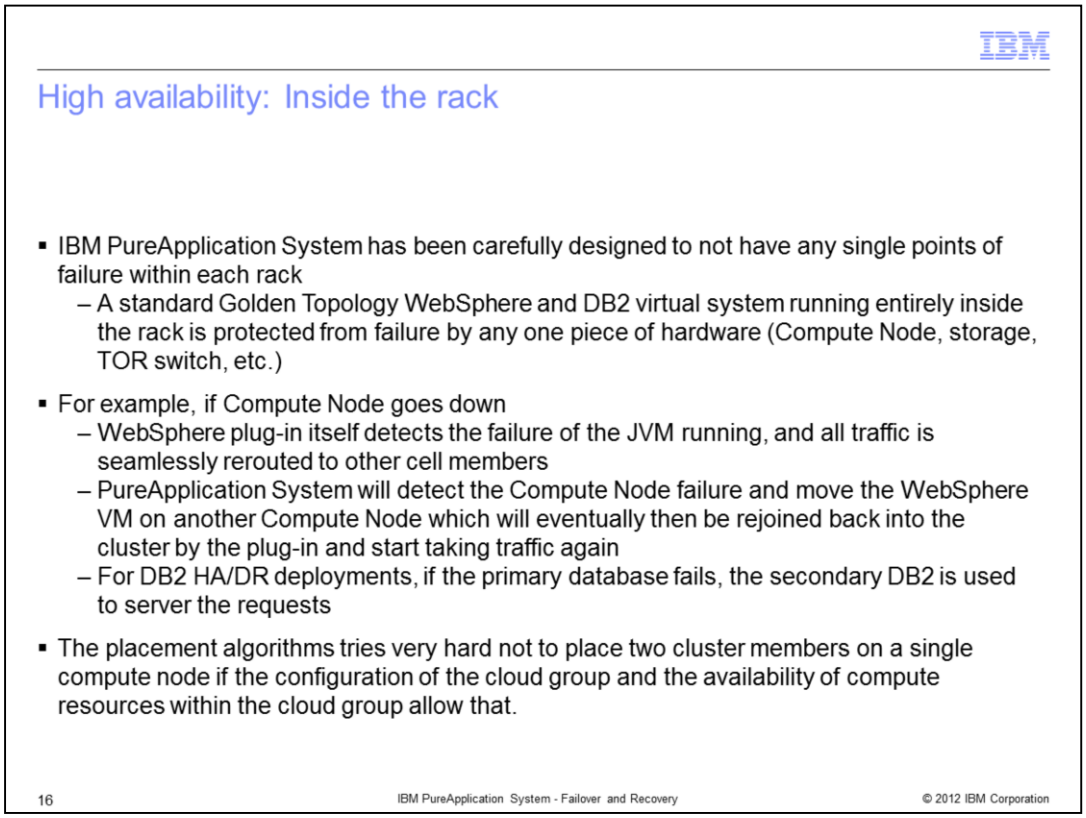
For the network controllers, there are redundant switches and cabling. Failure of 1 of the switch leads to reduced bandwidth, however, the rack continues to function.

For the storage controllers, each controller has 2 canisters that can service all the traffic to the storage. If one of them fails, the other one handles the I/O. For storage, both SSD and HDD, they are configured in RAID5, plus 1 spare. Hence it tolerates 2 concurrent drive failures without data loss.

If a compute node fails, PureApplication System will try to move the virtual machine to another compute node that has free resource to accept the VM. The VM is moved within the compute nodes belonging to the same cloud group. This is called rebalance or workload evacuation. If limited resources available on other nodes to handle the extra load, the virtual machines are started based on their priorities. Appropriate messages and events will be displayed for virtual machines that can not be moved due to lack of space in other compute nodes.



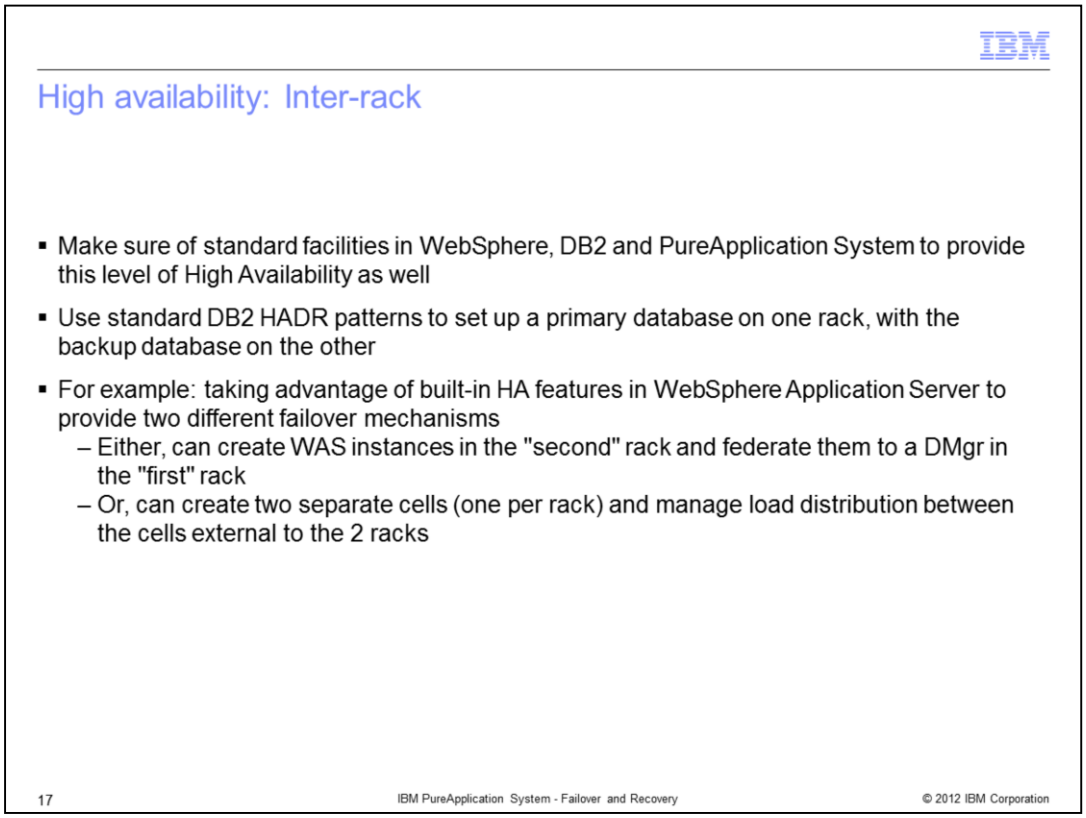
The next section covers some high availability scenarios.



Here, you first consider some of the high availability features inside the single rack. IBM PureApplication System has built-in high availability features inside the single rack and is carefully designed to not have a single point of hardware failure within the rack. For example, a standard golden topology WebSphere and DB2 virtual system deployments running entirely inside the rack is protected from failure by any one piece of hardware like Compute Node, storage, TOR switch, etc.

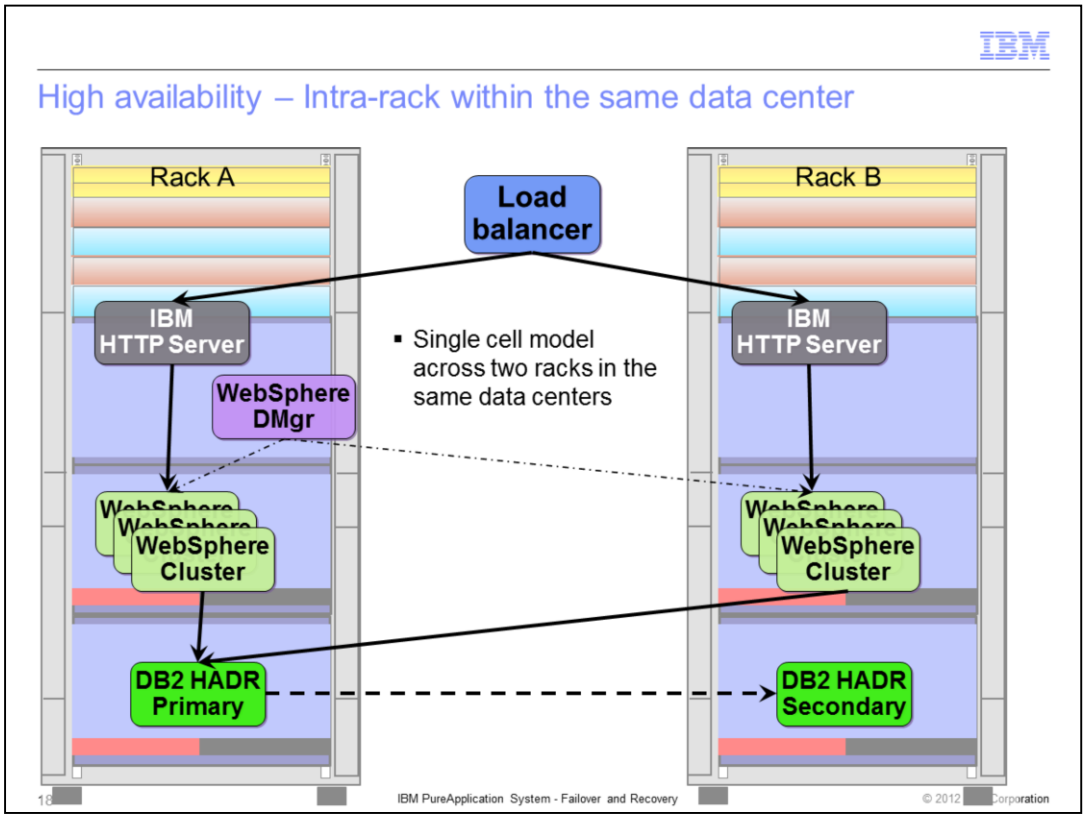
For DB2 HA/DR deployments, if the primary database fails, the secondary DB2 is used to serve the requests.

As an example, when a compute node goes down, several steps happen. The WebSphere middleware detects the failure of the JVM and seamlessly routes the traffic to the other cluster members within the WebSphere cell. PureApplication System will detect the Compute Node failure and move the WebSphere VM to another Compute Node , which will eventually then be rejoined back into the cluster by the plug-in and start taking traffic again. The placement algorithms of PureApplication System are intelligent enough that, in most cases, it tries very hard never to place two cluster members on a single compute node if the configuration of the cloud group and the availability of compute resources within the cloud group allow that. However, intra-rack HA does not help if the entire rack fails due to some catastrophic conditions in the data center. That is where inter-rack HA comes into picture. That's where you can again take advantage of standard facilities in WebSphere, DB2 and middleware to provide this level of high availability as well.



Here, you consider some of the high availability features across multiple racks. In this case, the standard middleware high availability features are used to support high availability across racks. Many IBM middleware like WebSphere, DB2, and others comes baked with high availability features that can be exploited in the same way it is done traditionally, outside PureApplication System.

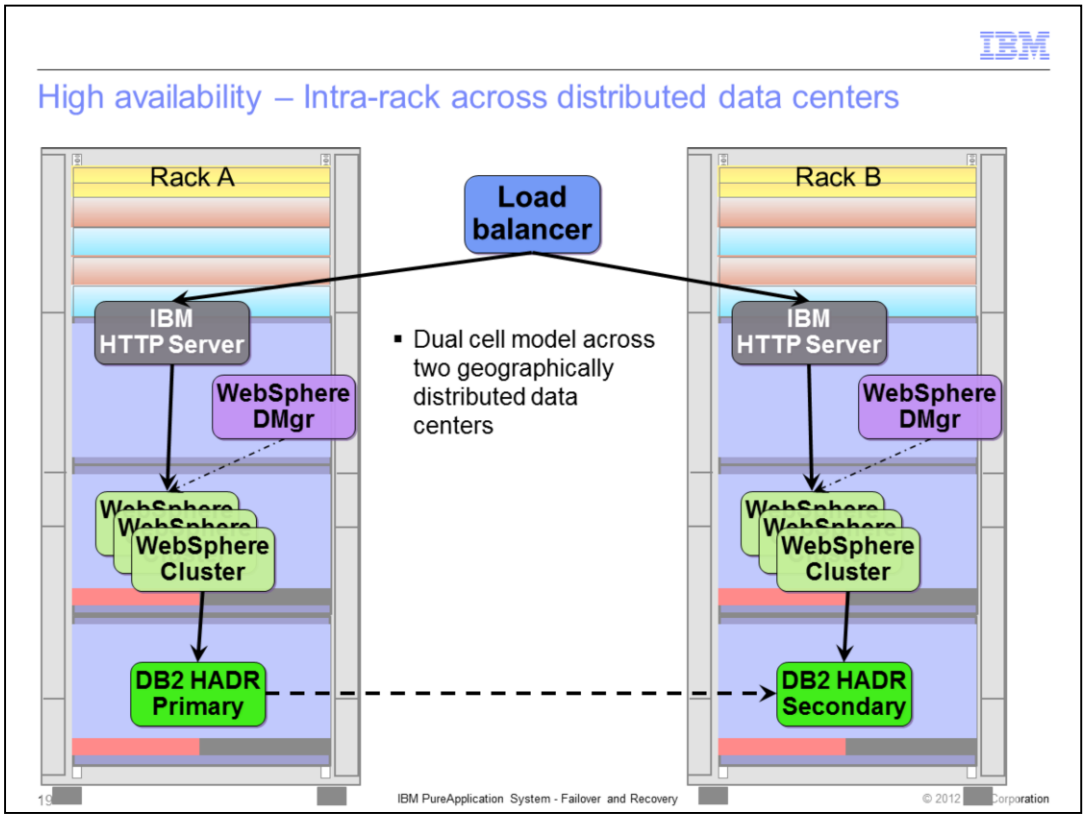
For example, taking advantage of the built-in HA features in WebSphere Application Server, one can provide either of the two different failover topologies. For mechanisms, either, one can create WebSphere node instances in the "second" rack and federate them to a deployment manager in the "first" rack, or, one can create two separate cells (one per rack) and manage load distribution between the cells external to the two racks. Some of these use cases are covered with more detail later in this presentation.



In this high availability intra-rack scenario, referred to as the "single cell" model, you begin by creating a virtual system pattern that defines a WebSphere network deployment cell. It consists of a Deployment manager, IBM HTTP Server (IHS) nodes, and WebSphere Application Server nodes in the first rack named Rack A. You then create a second virtual system pattern on the 2nd rack, named Rack B, that contained only IHS nodes and WebSphere Application Server nodes. You can specify the Rack A Deployment Manager on the Custom node cluster of Rack B and then deploy the Virtual System pattern. This defines the cell boundary as crossing both the racks. You likewise create a virtual system pattern for the primary DB2 HADR node in Rack A, and a second virtual system pattern for the secondary DB2 HADR node in Rack B. Note that in order for this to work, you need to configure an external load balancer to be aware of all of the IHS instances in the two racks. You also have to consider HTTP session management across the two racks. The simplest case in this approach is to enable database session persistence to the shared database.

In this configuration, you are now tolerant of a complete failure of either rack. If Rack A fails, then the IHS instances and WebSphere Application Server nodes on Rack B continues to take requests from the external load balancer, and the DB2 HADR secondary takes over from the failed primary node.

This model also allows you to provide WebSphere maintenance per rack one at a time, giving you the ability of WebSphere cluster member running at least on 1 rack while the other is undergoing maintenance.

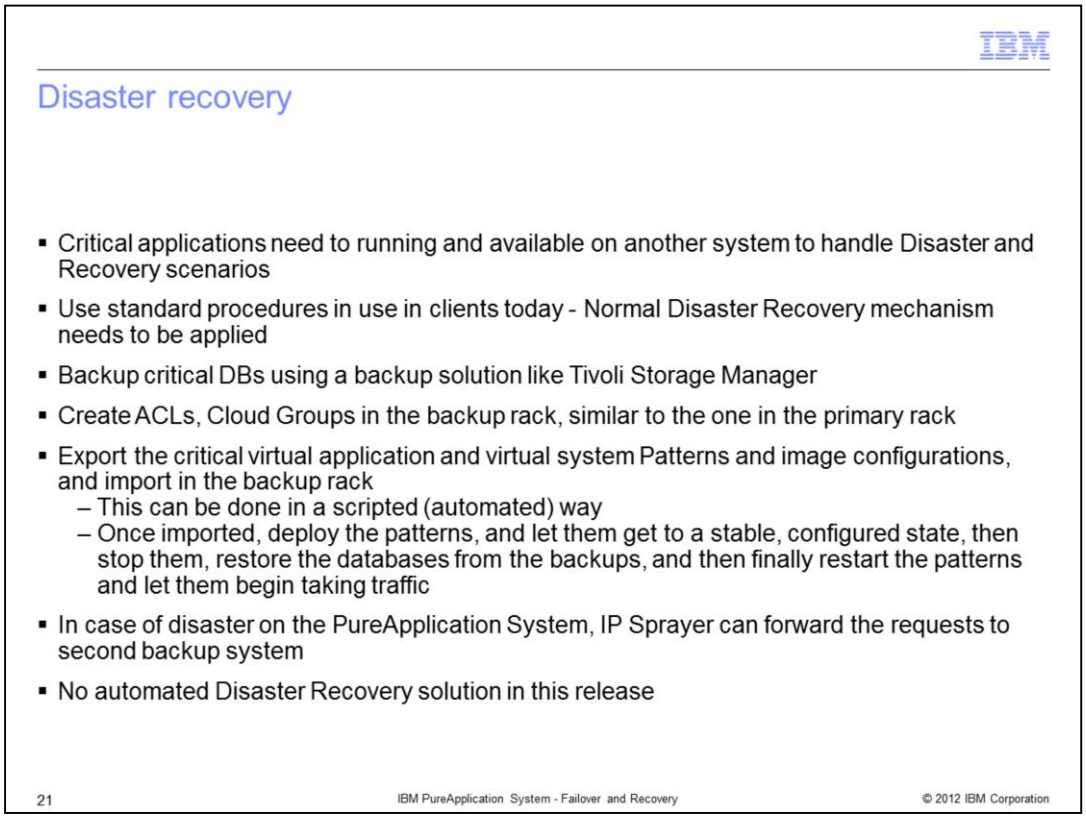


The case of two geographically separated PureApplication System racks is a bit more complicated. The communication necessary between the DMgr and its cluster members (for management, code distribution, and so on) is not efficient over a wide area network, and so it is not recommended federating cells across long distances as a best practice.

Hence the recommendation to create two cells as opposed to joining all the instances into a single cell as shown in the last example. This is referred to as "dual-cell" model, you need to create at least two different cells using a shared database, as shown here.

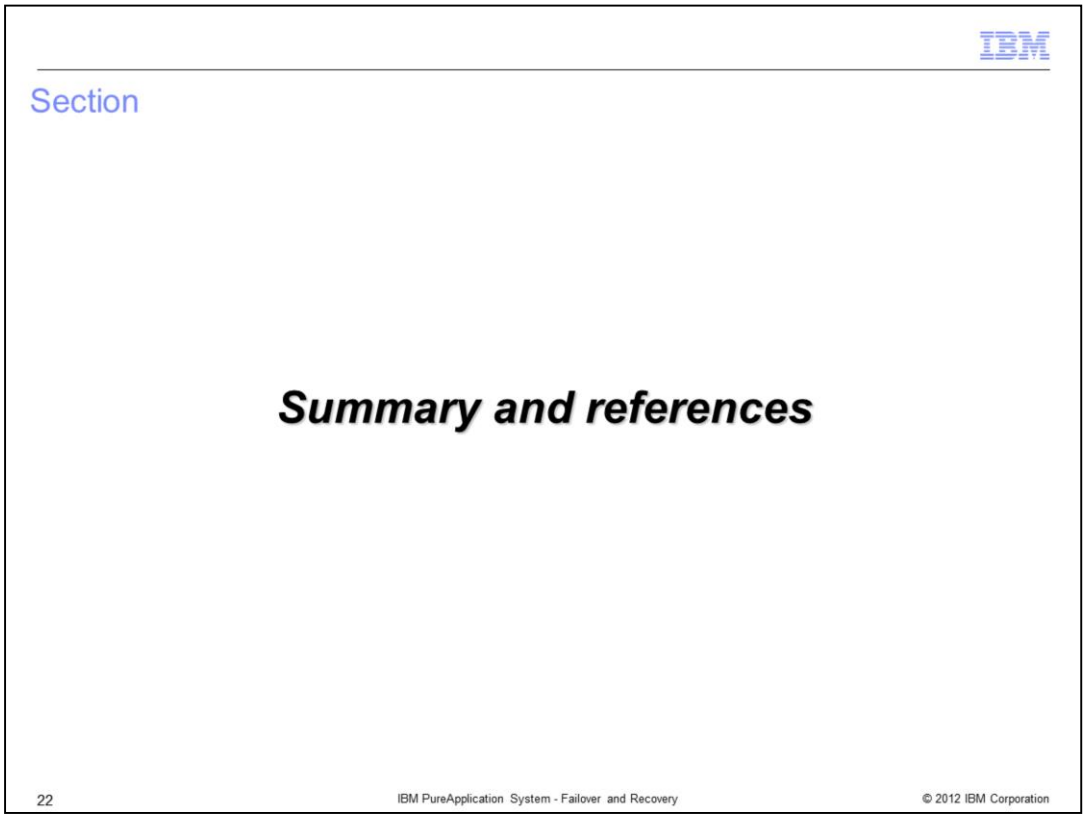
Using HTTP session replication across two cells through the shared database is possible, but it is rarely done. In most cases, session affinity is configured in the external load balancer. For example, requests for a session that was started in a particular cell will always be routed to that cell. If you can tolerate lost session data in cases of a failover, you can set up a separate local database for session persistence. The external load balancer is set up to feed traffic to the full set of IBM HTTP Server instances in both cells. If either rack fails completely, then the other continues to take traffic uninterrupted.

The next section discusses some of the disaster recovery steps.

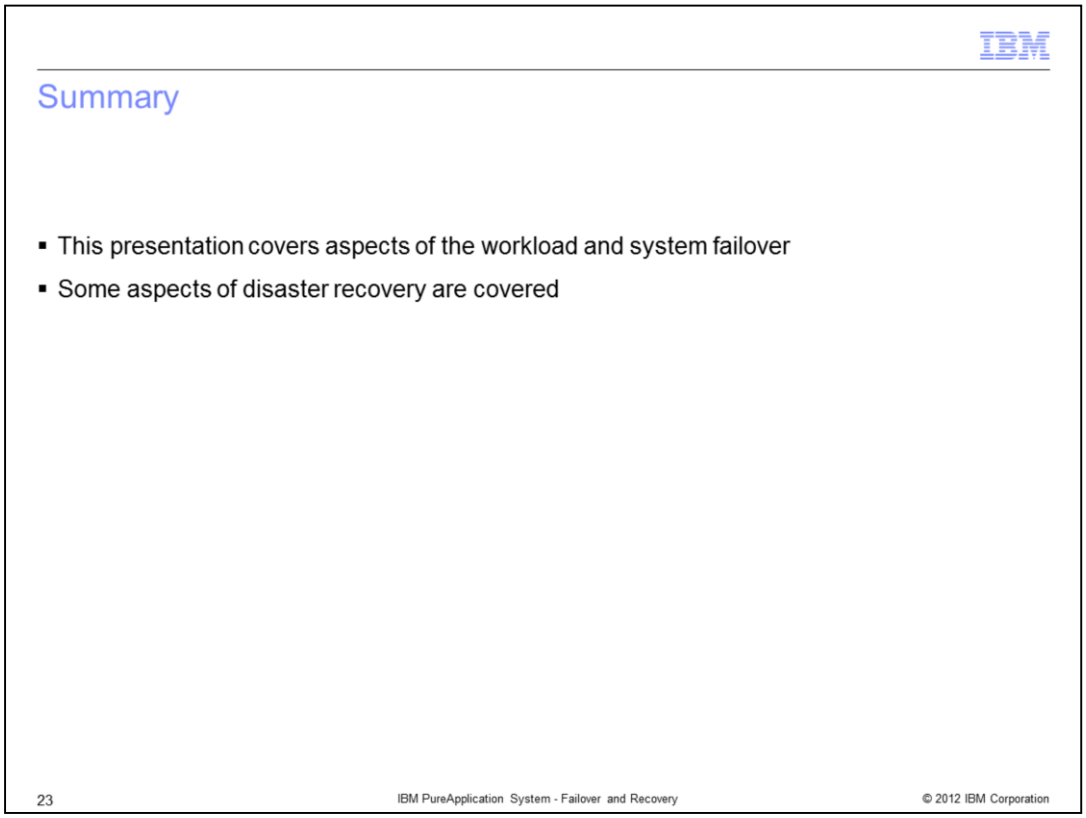


While in this release, there is no automatic process to create another rack for disaster recovery, there are some steps you can take to create another rack that contains critical applications either ready to run or already running in standby mode that can then used for disaster recovery. At the core, you continue to use the normal disaster recovery mechanism and backup the critical databases. The stand-alone DBaaS or DBaaS as part of Web Application pattern can be backed up using Tivoli Storage Manager. For DB2 VMs as part of virtual system patterns can use the standard backup mechanism supported by DB2.

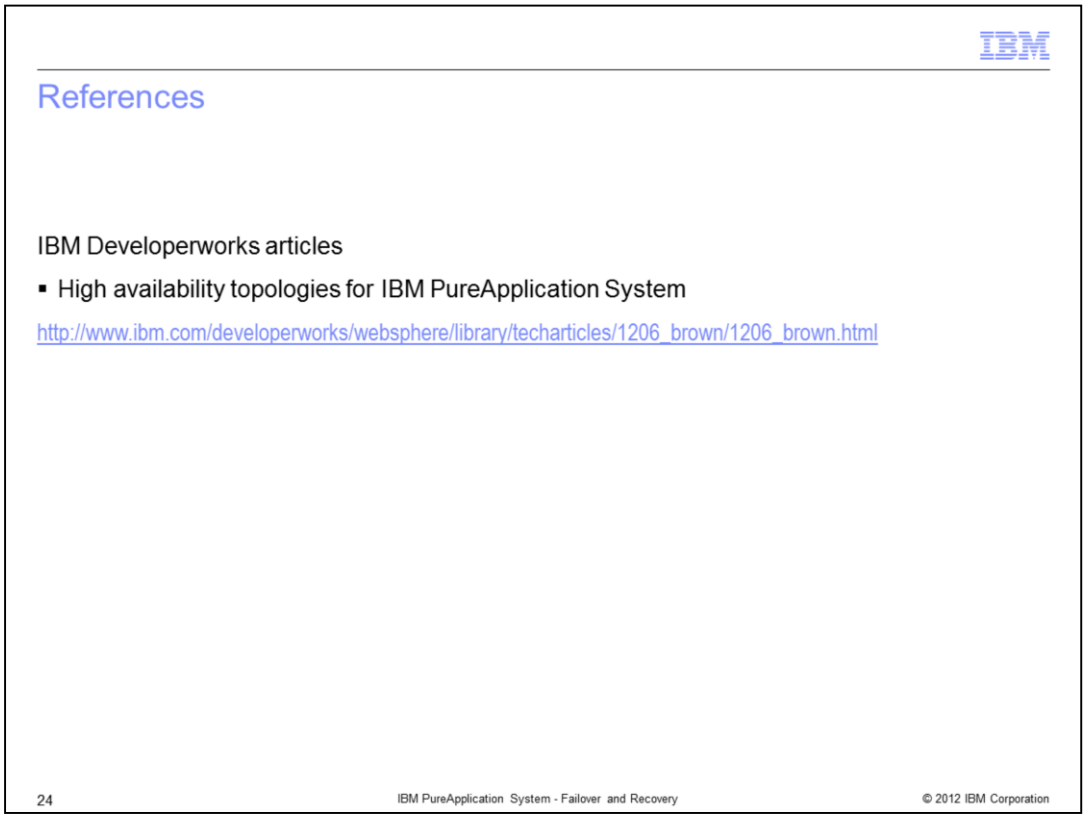
You create the same users and groups and ACLs, cloud configurations in the backup rack. Or, you can create the minimum set of user/groups, ACLs, cloud configuration in the backup rack. You then export the critical virtual application and virtual system patterns. The exported patterns can then be imported to the secondary rack. One can use an external sprayer to forward the request. When the initial rack is down, the sprayer can forward the request to the backup system.



The next section covers the summary and references.



This presentation covered the Workload and System failover use cases from the workload and System failover, along with some aspects of Disaster Recovery.



Listed in this slide are helpful references and resources.

Trademarks, disclaimer, and copyright information