



High Availability

A Whitepaper

by PaperCut Software – June 2018



Contents

Introduction	3
Why High Availability is a good thing	3
Organizational factors	4
Business value	4
User behavior	4
System infrastructure	4
Achieving High Availability	4
Redundancy	4
Recovery	5
What is the right level of High Availability?	5
Recovery Point Objective (RPO)	6
Recovery Time Objective (RTO)	6
Protecting PaperCut with High Availability	7
PaperCut examples	8
PaperCut's scalable architecture	8
PaperCut Application Server	8
PaperCut Site Server	9
Print providers and print servers	9
External database	9
Network Load Balancers in detail	9
Where to use an NLB	9
How does a Network Load Balancer help with printing?	10
Does network load balancing work with Find Me Printing?	12
Virtual machine clustering in detail	14
What can Virtual Machines do?	14
Defining your Virtual Machine clustering environment	15
Defining your Virtual Machine clustering setup	15
Microsoft Failover Cluster Manager in detail	16
What is Microsoft Failover Cluster Manager?	16
How to set up Microsoft Failover Cluster Manager	16
Resiliency and redundancy considerations	17



Introduction

This white paper presents the PaperCut philosophy on providing High Availability to our components and the surrounding print infrastructure. It's intended for IT decision makers and anyone evaluating HA for their PaperCut installation.

When people talk about **High Availability (HA)**, they're usually referring to the ability to use a computing service without noticeable interruption. HA generally provides a significant increase in the end-to-end availability of system functionality when compared to non-HA systems. However, they're often much more complicated to deploy and maintain. The question we need to answer is: which systems need to be protected with HA techniques, and at what level?

The concept of HA can be found in all sorts of things besides computing services. For example, we want our automobiles to be highly available and we take steps to protect some of the features that could fail, such as tires. This is why most vehicles carry a spare tire, but since there are costs to carrying these spares, we generally only have a single spare tire. HA always has costs in time and money.

It would be fabulous if computers never crashed or suffered outages from hard disk failures. But they do. And even if they didn't, there'd still be downtime from routine maintenance, human error, hackers, network connectivity, natural disasters, etc.

So considering which systems to protect – and at what level – is important. When it comes to our computing services, we don't want the equivalent of carrying six spare tires when one will do quite nicely.

Why High Availability is a good thing

HA protects resources so that in the event of a computer service outage, your business can continue with minimal interruption or decrease in throughput. By reducing the risk of downtime, we improve business continuity.

One way that HA reduces the risk of downtime is by eliminating single points of failure. If your entire PaperCut system runs on one server, then that server is a single point of failure – and so are the components of that server like the hard drive, power supply and network interface card, by the way. Even software such as driver updates and operating system patches can bring down the system if they contain serious defects.

All in all, HA gives you methodologies to eliminate single points of failure, reduced risk of downtime, and increased business continuity. *That's* why it's a good thing.

Organizational factors

Print Management software is commonly used to provide many of the traceability and security features described in this whitepaper. For example, secure print release is an important feature of any print management system. PaperCut NG or PaperCut MF can support all the security measures described in this document.

Business value

First, we must know what parts of the business need to continue, and what their value is to the overall organization. If we're running a hospital and the snack vending machine can't process payments, it's probably not as vital as ICU patient monitoring. However, if you're a university during finals week, that same vending system could be mission critical.

User behavior

Another impact to proper HA planning is user behavior. Are there peak usage times which stress certain components of the computing system? Do some applications create higher system load? Which users will need system access, even in the event of a disaster?

System infrastructure

Are there multiple types of devices that need to be protected? Have we identified all the pieces of the system? What if we lose power to the primary datacenter?

Textbooks are written on techniques to protect computing services. Whether it's the servers, operating systems, databases or power sources, we need to understand business goals and any single points of failure in the computing systems that will put them at risk.


Achieving High Availability

Achieving HA is accomplished primarily through two methods: **redundancy** and **recovery**. Both give the computing system resilience (i.e. an ability to return to full operation).

Redundancy

Redundancy solves the problem of a potential failure by having a duplicate standing by. It uses technologies like RAID, [Virtual Machine \(VM\) images](#), [clusters](#) and [Network Load Balancers \(NLB\)](#).

If you're using RAID and a hard disk crashes, no problem – the data's been redundantly written to other disks. If you're using virtual machines and the whole VM crashes, no problem again – just spin up the latest VM image on a new VM server and you're back in business. Clusters and NLBs have multiple servers



running and can divert computing requests away from a failed server to one that's still up. Theoretically, you can have an extremely low risk of downtime by implementing redundancy.

However, the cost is usually at least double for a redundant system. Plus, there's added time and complexity to build and maintain these systems, which also require highly trained personnel.

Serious consideration should be given to determine the type and level of redundancy. One PaperCut customer had a very sophisticated Linux HA system with clustered DNS servers and clustered database servers. It was bulletproof. However, when their system administrator left the organization, no one knew how to maintain it. System upgrades were painful and time-intensive. Eventually, they scrapped the complex, over-engineered system and implemented VM servers that could be spun up on new hardware at a moment's notice. It provided the same level of redundancy, with much less complexity.

Recovery

HA can also be accomplished with a good disaster recovery plan. This is the most basic form of HA and avoids most of the complexity of many HA techniques. Good disaster recovery plans will have procedures to minimize downtime for key systems. One such procedure could be taking a database backup every night, and writing daily transactions to an offsite server. This should give you the ability to have the full database back online within a short time, even if the main database server crashes and burns.

What is the right level of High Availability?

Speaking of burning, we had another PaperCut customer with a robust installation that included clustered application servers, clustered print servers, and clustered database servers – all of which pointed to a SAN. From a system point of view, it was on the high end of HA... Until a fire tore through the datacenter, and then it wasn't on the high end of anything, let alone available. Forced into a redesign, they reconstructed their PaperCut installation and opted for more modern virtual machine technologies to provide the same level of HA.

This leads us to a primary consideration: what is the right level of HA? There is a significant difference in TCO between providing 99% uptime and 99.99% uptime. Is the difference necessary and worth the cost? Even if HA is a good thing for your business, and the ways to achieve it are well understood, the question still needs to be considered: what level of HA is necessary? You'll have different answers to this question for various functions in your organization. Your mission-critical systems need more HA techniques for more parts of the infrastructure if the cost of an outage would be greater than the cost of providing HA.

For example, vital systems such as order placement, user authentication and database may need 99.99% uptime. This might be enabled with multiple HA techniques like virtual machine snapshots, clustering, synchronous replication,

hot sites and off-site backups. However, the print system might be just fine with an hour of downtime.

Our focus when considering HA should be on two objectives:

Recovery Point Objective (RPO)

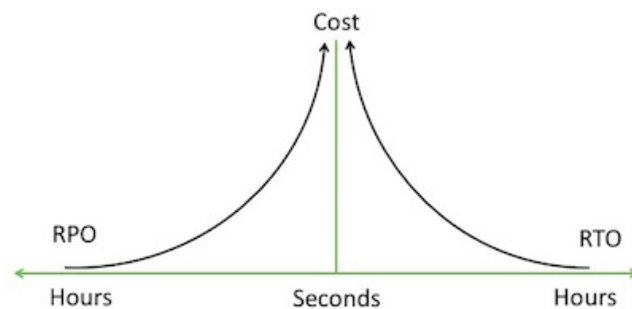
RPO is the length of time between taking snapshots of your data. It's a measure of how far back in time you must go to get a recovery point. It's also the amount of time where the business process can cope with a loss of data.

Recovery Time Objective (RTO)

RTO (aka Mean Time to Recovery) is the maximum tolerable time from point of failure back to normal operation.



RPO and RTO need to be carefully considered, because together, they determine the cost to recovery. The smaller the times for RPO and RTO, the larger the cost to recovery. If you want recovery points measured in seconds and recovery time in minutes, then expect a very high cost to recovery.





Protecting PaperCut with High Availability

Now let's apply all this to a PaperCut deployment. We don't want to start with the assumption that printing systems need to be protected at the same level as other business functions. We should start with the business objective questions. What's the maximum amount of time that print jobs (and their accounting data) could be lost, and need to be reprinted (RPO)? What's the maximum time allowable from failure of printing systems, to full recovery (RTO)? And third, what's the expected total cost to recovery?

It's possible with PaperCut to improve overall system resilience at multiple points in the infrastructure. PaperCut recommends using HA technologies and methods that are most familiar to the customer, and where they have trained personnel to support them. This reduces overall system complexity by avoiding introducing new tools and procedures to learn and implement in the event of an outage. This is a primary reason why PaperCut doesn't mandate HA methodologies or create HA products of our own.

Remember our clustered Linux HA customer mentioned earlier? It turns out that their single point of failure was the person who set up the complex environment using multiple HA technologies that no one else understood. PaperCut doesn't want to force customers to add yet another tool on the HA stack.

HA technologies have evolved to the point where very favorable RPO and RTO can be achieved without adding this cost and complexity into PaperCut products. The customer shouldn't have to learn our way of providing HA; they should be able to use what they already know. Even if we wanted to build all this in to PaperCut, we couldn't fully protect against the most common sources of downtime: full hard disks, dead NICs, and human error. The full printing system, including PaperCut, can be protected against downtime with off-the-shelf HA technologies.

Therefore, the discussion to add HA for PaperCut should start with, "How does the customer defend against downtime on other servers, and in particular the printing systems?" For example, it would do us little good to defend the PaperCut server and not other print servers as well. Anywhere that PaperCut is running, or where it utilizes a resource, should be considered in the HA plan. Simply put, the main pieces that we want to evaluate protecting are the PaperCut Application Server, PaperCut [Site Server](#), Print Provider (i.e. print servers), database and the multi-function devices (MFDs). If a customer already employs HA methods on other servers and resources, they should be able to include PaperCut in the same manner with very little additional training or expense.

Whatever methods you employ for PaperCut HA, "The general principle is start light, and build over time" (Chris Dance, PaperCut CEO).

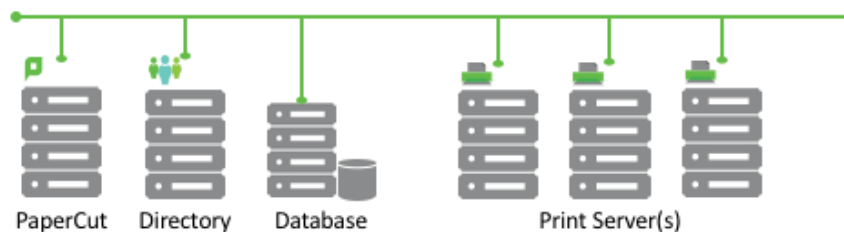
PaperCut examples

There are several proven techniques to provide HA, each with differences that may impact your decision of which one(s) to use. There are also technical constraints that may make some techniques unadvisable (e.g. using an NLB for SQL databases). The overarching principles are:

- Use the tools with which you have a depth of experience
- Provide the level of HA that meets business objectives
- Start light and build over time

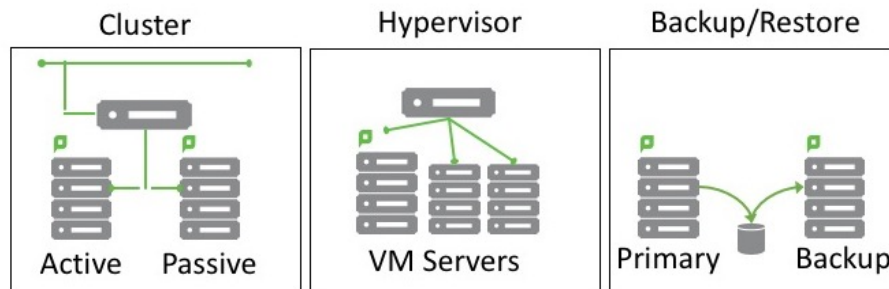
PaperCut's scalable architecture

PaperCut is designed to be a scalable system. This means that critical components (e.g. application server, database) can be located on separate servers to allow for growth and performance improvements. This also has the advantage of making these components more easily protected for HA. The diagram below shows a very common deployment with a PaperCut Application Server, a user directory, a database server, and multiple print servers. In this example, it'd be almost trivial to protect the PaperCut app server, since the critical data is in the database.



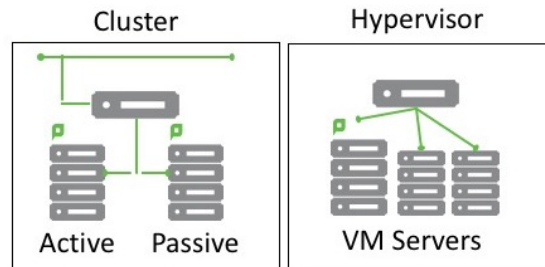
PaperCut Application Server

The **PaperCut Application Server (app server)** can be protected with clustering, virtual machines, or simple backup and restore.



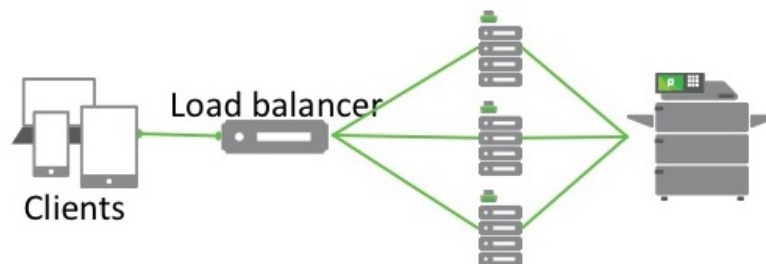
PaperCut Site Server

The **PaperCut Site Server (site server)** provides resilience to the printing system if the connection to the app server becomes unavailable. The site server itself should also be protected with clustering or a virtual machine.



Print providers and print servers

The **PaperCut Print Provider** runs on print servers and can employ all the methods mentioned above and **Network Load Balancers (NLB)**. An NLB distributes application traffic (print jobs in this case) across a number of servers. On top of load-balancing print jobs, an NLB will bypass a server that isn't responding, which adds protection for a failed print server.




External database

An **external database** can be protected with clustering and VMs, as well as some database-specific techniques, such as synchronous replication and off-site transaction logs. Check with the customer and the database manufacturer for the best choice.

Network Load Balancers in detail

Where to use an NLB

As described above, an NLB basically divides network traffic (i.e. service requests) among multiple servers that can respond to the demand from clients. One main



advantage is that all the clients make their request to the NLB, not the individual servers. This technique is used extensively in the internet. The most practical way for web sites to respond to increased load is by utilizing an NLB. Otherwise, we'd all need to browse to google2.com if google.com was busy – and if they're both busy, we randomly try google42.com. Not fun.

Using an NLB allows everyone to browse to the same internet site (e.g. google.com) because the NLB decides which actual server is available to respond. Now the term NLB makes more sense; it literally balances the network load across multiple servers.

The NLB is a very useful tool in several situations. We've mentioned balancing load, but it also allows for easier maintenance of individual servers. If you have sixteen servers behind your NLB and one of them needs a new disk drive, you can remove it from the NLB, replace the drive and then put the server back in the NLB.

Another feature of NLBs is protecting against an individual server failure. Using the scenario above, what if that bad-mannered disk drive crashed before you got around to replacing it? You still have fifteen other servers to bear the load. Disaster averted.

However, there are some situations where an NLB would cause problems. Internet sessions are designed to be "stateless." This means that the server doesn't keep track of the state of the client session. So, if we browse to google.com, we might get connected to google7.com to let us login, and then be connected to google166.com to prompt for our search, and then google99.com when we click a link. These stateless internet sessions work well with an NLB.

On the other hand, "stateful" connections keep important data about the session in server memory. If we put an NLB in front of stateful servers, it'll confuse them to death. Stateful server connections are more efficient than stateless ones since less data needs to be exchanged between the client and server.

The PaperCut Application Server, for example, is stateful; it keeps track of connected clients to make communicating more efficient. So, the app server can be protected with clustering, virtual machines or backup and restore, but not an NLB.

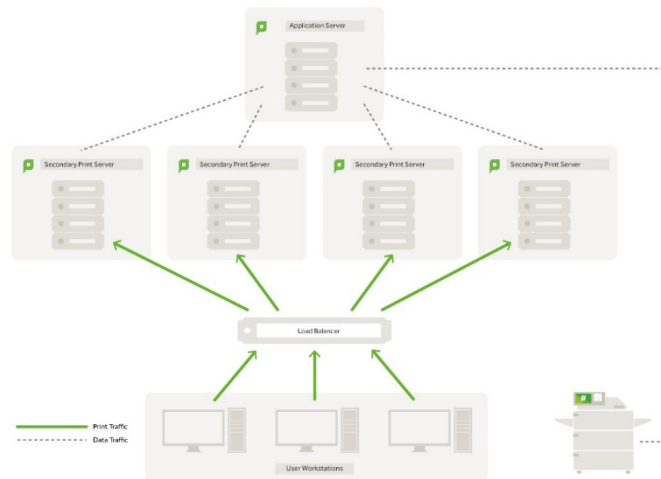
How does a Network Load Balancer help with printing?

Just as a web server farm can allow multiple requests to be split across multiple web servers, we can achieve the same effect with print jobs. It's possible to configure a load balancing device (e.g. F5, NetScaler) to accept incoming print requests and split them across multiple print servers.

NLBs can be useful in a customer environment where there are large numbers of users constantly printing, or where there are large print jobs being frequently produced. Load balancing gives system administrators an easier method for print administration, and a flexible approach to the print environment. There are a couple of different ways in which PaperCut can operate with these devices.

As you might already know, PaperCut can be installed as a primary server, secondary server and a site server. The primary server is the installation of the main PaperCut application, which is where the software is managed.

This is what the basic setup for load balancing will look like:



The **PaperCut secondary print servers** are installed behind the NLB.

Each PaperCut secondary print server is connected to the app server and will be configured with all available print queues.

For example, if you have five printers in the network (Printer1, Printer2, Printer3, Printer4, Printer5), then each PaperCut secondary print server would have a print queue for each of these five printers.

The NLB can be configured to accept print traffic from client machines, and redirect them to any PaperCut secondary print server.

In most cases, you'll configure a DNS name that references the IP address of the NLB.

Clients will use that DNS name to reference the printers. For example, if we created a DNS name for the NLB of 'NLB-Uno', clients would connect to NLB-Uno\Printer1 and NLB-Uno\Printer2.

When a user prints to one of these print queues, the job will be redirected by the NLB to one of the PaperCut secondary print servers. And this means that the job may end up on any of the four PaperCut secondary servers (depending on how the NLB has been configured).

The app server will see each of the PaperCut secondary servers and the associated print queues individually, and will release the job to the appropriate printer as required.

If [secure print release](#) is being used, the job will be redirected by the app server to the correct printer when the user logs on to the device. Otherwise, the job won't be held at the server and will be directly sent to the printer.

Does network load balancing work with Find Me Printing?

It sure does. If you're using [Find Me Printing](#), you need to configure an additional queue on the PaperCut secondary print servers. This queue will be your virtual Find Me queue. Make sure you replicate this across all PaperCut secondary servers.

The end user's queue would be NLB\Find Me. The NLB can be configured to redirect the incoming Find Me queue to one of the four print servers.

From the app server, you'll see four Find Me queues, and PaperCut will report on each one individually. Since each queue is a separate instance, we can take our solution one step further and configure server affinity for the print queues. This ensures the server that receives the print job will be the same server that sends the job to the printer.

We configure server affinity because it reduces network traffic. Basically, it stops jobs from being received by one server (e.g. Server1), and then transferred to another server (e.g. Server4). You can see what happens to a print job with and without server affinity below.

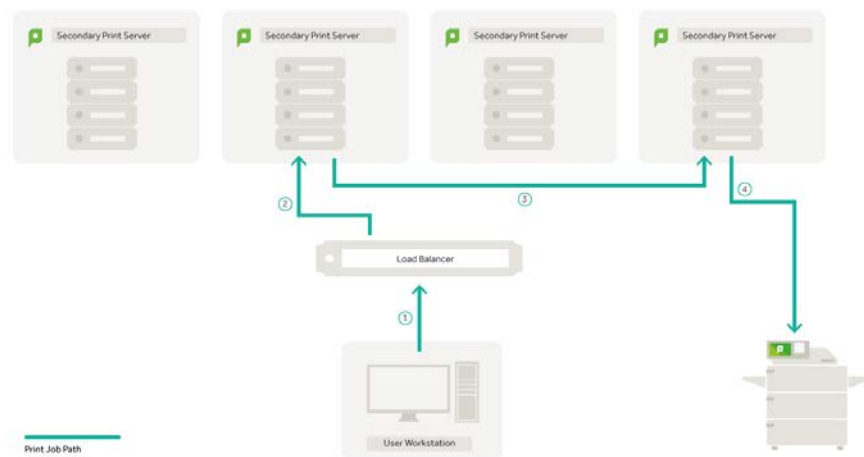


Figure 1 Without server affinity

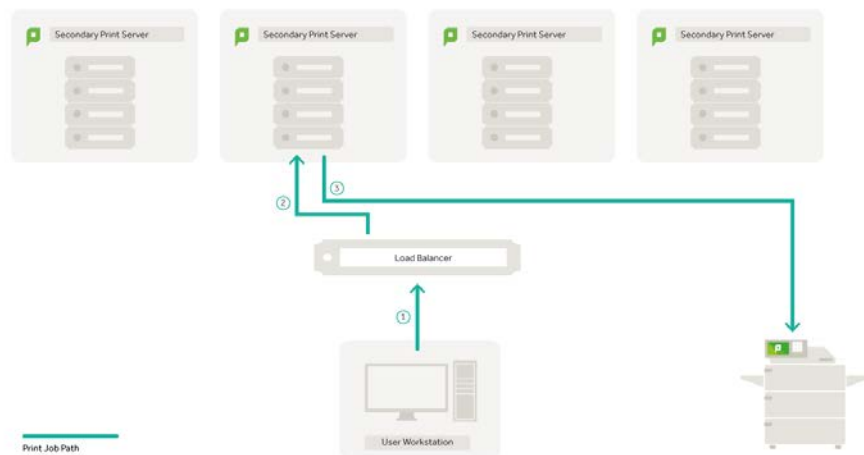


Figure 2 With server affinity

To configure server affinity, there are a few steps to complete in the PaperCut Admin Interface:

- Log into the PaperCut Admin Interface
- Go to 'Printers'
- Choose one of the Find Me queues (e.g. Server1\Find Me)
- This must be configured as a virtual print queue
- In the Job Redirection settings, choose the printers **on this server** that you want to be able to release jobs to (i.e. from the Find Me queue).

In our example, we'd choose server1\printer1, server1\printer2, server1\printer3, server1\printer4 and server1\printer5

- Save the settings
- Now, jobs sent to the Find Me queue on server 1 can only be released to a print queue *on this server*
- Repeat these steps for your remaining Find Me queues, ensuring that you only select the output queues for each relevant server.

Now that PaperCut will only release print jobs to queues on the receiving server, we need to configure each device to check all four Find Me queues:

- Go to 'Devices'
- Choose a Device where you want to use Find Me printing
- Scroll down to 'Print Release' and make sure this option is selected
- In the queue list, select the four Find Me print queues

- Save these settings and repeat for the remaining devices

Now each device will check the Find Me queues on all four servers to look for print jobs.

From the end user's perspective, they see a single print queue when they print. When they release their print jobs, it all appears to be a single print queue as well. However, their print jobs could all be on one server or spread across multiple servers.

This is a great solution to provide redundancy against a single print server failing. It also allows for a scalable and easy to manage solution.

NOTE: The PaperCut Application server cannot be load-balanced. Jobs can only be load balanced across PaperCut Secondary Print servers.

Virtual machine clustering in detail

What can Virtual Machines do?


Virtual Machine (VM) technologies, such as VMWare and Microsoft Hyper-V provide great flexibility in deploying servers within an organization. VM implementations can also provide HA using VM clusters. When a VM is running in a highly available VM cluster, any failure of the physical hardware has minimal impact on the running VM, which is simply transferred to another node in the cluster.

Implementing HA using VM infrastructure is a viable alternative to the built-in operating system and application clustering support. This allows you to set up PaperCut in the same way you would on a physical server, but allows the VM infrastructure to provide the HA.

Clustering at the VM level offers advantages over other traditional clustering setups, such as:

- Your software, drivers and settings only need to be installed and configured once in a single VM image
- Depending on your VM infrastructure, when a physical node fails, the VM can be shifted to another node with little downtime
- Simplified backup processes
- Disaster Recovery (DR).

VM hypervisors detect when a VM becomes unresponsive. You should consider whether you'll augment this with application-level monitoring. Although the VM might be running normally, the underlying application may have problems and



application-level monitoring can detect this. Ways to perform application-level monitoring include:

- Loading an Application Server URL to test the server is running
- IP pings
- Checking that PaperCut services are running.

Defining your Virtual Machine clustering environment

There are many VM deployment strategies you can leverage, depending on the VM platform you're using. This includes VMs hosted in different physical boxes, or even in different sites. This also offers DR options.

When selecting a VM product, it's particularly important to consider the following features:

- Fault Tolerance (FT) – provides continuous availability for VMs by automatically creating and maintaining a secondary VM identical to the primary VM
- High Availability – lets you make a new VM available to minimize downtime if an existing VM fails. Generally, FT is a functionality added on top of HA that provides seamless switch-over if there's no loss of state.
- Application HA – lets you make a new VM available to minimize downtime if the application fails. It's important that your VM product offers application-level monitoring.
- Data replication and backup – lets you backup and restore in-memory and application data. Replication of in-memory data in case of FT or DR features might also play an important role. Specific proprietary algorithms for the replication of memory segments usually reduce the bandwidth needed and are very efficient.


Defining your Virtual Machine clustering setup

There are many ways in which you can deploy PaperCut on VM infrastructure. Consider the following for implementing a VM-based clustered PaperCut installation:

Mode 1: Clustering at the print layer

Configure print servers as required in your VM environment and configure them for HA. Install the PaperCut secondary server to monitor printing on the print servers.

Mode 2: Clustering at the Application Server layer



Configure a new server in your VM infrastructure to host the Application Server. Configure this VM with HA. Install the PaperCut Application Server. You can then choose to set up your print servers for HA.

Microsoft Failover Cluster Manager in detail

What is Microsoft Failover Cluster Manager?

Since Windows 2012, Microsoft have reconfigured how you can manage HA with print services. They introduced the ability to use Hyper-V and failover clustering to make your print server a highly available VM. This solution provides full server failover options and can be implemented with the PaperCut servers.

How to set up Microsoft Failover Cluster Manager

If you'd like a nicely detailed article on setting up PaperCut servers on Microsoft Failover Cluster Manager, check out the [Microsoft Failover Cluster Manager](#) section in the PaperCut manual. Otherwise, keep reading to get a breakdown at a higher level.

System requirements:

- Two or more physical servers
- MS Server 2012 or 2016 with Hyper-V
- iSCI SAN with 2 drives configured
- Drive 1: 5GB (to be used for the Quorum if using only two nodes)
- Drive 2: VM server storage.

You'll also require the following:

- Administrative rights to join machines to the domain
- SAN IP address
- IP Addresses for the following:
 - Physical Servers
 - the Cluster
 - the VM

Setup step-by-step:

Step 1 – configure roles / features on nodes for HA

Step 2 – connecting to iSCSI network drives

Step 3 – create the failover cluster

Step 4 – create a HA VM

Step 5 – set up your print server

Site Server resiliency in detail

Resiliency and redundancy considerations

For customers with distributed deployments, considerations such as redundancy and resilience to network outages are often a high priority.

A robust solution should defend critical points of failure, allowing an organization to continue operating while a network is under duress. For PaperCut, this means a robust deployment should ensure the availability of printing over unreliable network links.

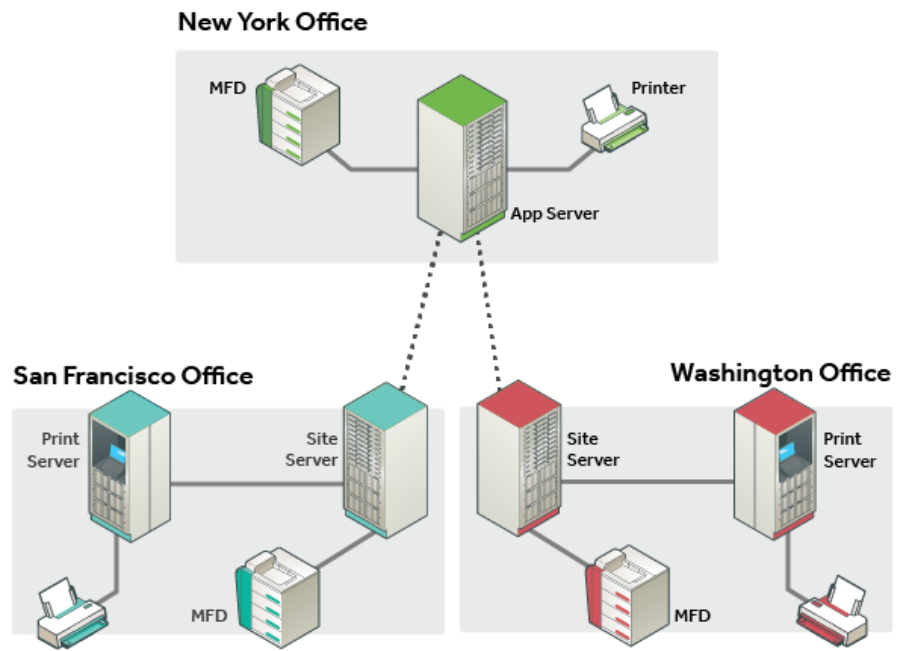
The installation of PaperCut Site Servers gives customers peace of mind because access to printing resources won't be interrupted by unexpected network dropouts.

The Site Server duplicates the key features of a PaperCut primary server to a local site during an outage. MFDs are configured to connect to a Site Server as if it were the primary server to remove their reliance on WAN links. PaperCut secondary servers (Print Providers) are also aware of their local Site Server, providing a failover server if the primary server can't be contacted.

This simple but effective design delivers HA to MFDs and support for Secure Print Release, including Find Me printing.

The Site Server installs in minutes with minimal configuration steps and no ongoing administration. Installers and Administrators need no specialist skills in database management or replication to provide business continuity. The Site Server ensures it's kept up to date with the current state of the primary server, transparently performing the role of the primary server when needed. Once a connection to the primary server can be re-established, the merging of local Site Server logs and transactions back to the primary server is also seamlessly managed by the Site Server.

The close relationship between the Site and primary servers allows the support of the same set of operating systems and databases for installations. It's perfectly valid to promote an existing PaperCut secondary server to a PaperCut Site Server to improve a site's resiliency.





Thank you

Author

David O'Hara (Lead Solutions Architect, PaperCut Software)

PaperCut HQ

sales@papercut.com

papercut.com

Support

support@papercut.com

papercut.com/support