Server Migration Overview (Page 1 of 3)

By: Tom Weeks Corporate Technical Trainer - Rackspace Managed Hosting [view other articles by this Author]

Not Helpful Partially Helpful Helpful Very Helpful Rate Article Average Rating: ★ ★ ★ ★ (Very Helpful)

Migrating your site and configurational data from one hosting provider to another can be a hair-raising experience. There can be many issues that can throw a real wrench in the works of your migration Dealing with each of them is important to ensure your data and domain move happens quickly, smoothly, and without web or mail outages.

The following overview looks mainly at the migration of UNIXor Linux based Internet Web/mail servers; although many of these same issues apply no matter which OS you run.

Leaving Your Existing Server Hosting Provider
Moving from one hosting provider to another can be a tricky job. Some providers do things such as maintaining control over your domain names, or will try to prevent you from moving your domains to another provider. FUD (a combination of fear, uncertainty, and doubt) is one of their best tools; they may tell you that you'll lose much of your domain's e-mail, that you'll have to re-purchase all of your SSL certificates, that your sites will be down for a week or more, or that you'll have perpetual problems with web and file-based permissions, as well as FrontPage. While these tales have some basis, there are technical ways around each issue.

Web Server Issues
Some people prefer to migrate to a fully patched system. Others prefer to move to an identical server (software-wise). Either way, the software and server daemons on the source and target servers make up a major part of the prerequisites list. In short, you need to do your homework and make sure that the data you are moving does not have application version reliance that could make things difficult for you later (e.g. perl vs. mod_perl, php versions, etc). From a bird's-eye view, some of the technical issues that you need to check on the target server before the move begins are:

- OS install and supported libraries (e.g. Linux and glib version)
- Version of apache (or web server service used)
- Version of Sendmail (or mail service used)
- Module, Perl and CGI engines installed and configured
- BIND/DNS software
- Same patch levels (old Vs new) of most content-sensitive binaries
- Same or greater hardware configuration
- Same or greater security/access rights (TCP-Wrappers, ACLs, etc)
- Any software or configuration dependencies on your IP address
- Any commercial packages that register their key based on IP, FQDN or servername
- Roughly the same OS and filesystem UID/GID (user/group) permissions

This last point is usually one of the trickier ones. If you're on a Linux/UNIX, server, get a copy of your entire old /etc directory moved over to a secure location (probably the /root directory) on the target server for quick reference. Then recreate the users on the new system with the same UID/GIDs if possible (more on this below).

If the source and target OS versions are close enough, and you can verify that the system crypt/hash functions are the same (i.e. the passwords are encrypted using the same algorithm), you may be able to copy/paste parts of the old /etc/passwd, shadow, and groups files over to the new system to help match up user and group ids (UID/GIDs) - sometimes even keeping the system and user passwords intact! Just remember NOT to copy entire /etc config files over the target machine's config files. You only want to copy/paste the regular users (not the other system accounts like daemon, nobody, and ident). Doing so is just begging for major headaches. Usually, on Linux systems, regular user accounts are those with a UID number of 500 or higher in /etc/passwd:

```
bsmith:x:  503:  503:  Bob Smith:  /home/bsmith:  /bin/bash
^^^        ^^    ^^    ^^^         ^^^^^          ^^^^
user       UID   GID   name        home dir       login shell
```

The filesystem and movement of data can be the hardest parts of a migration, because every file in the system maps to a specific user/UID and group/GID (not even mentioning special file links, security settings, etc). Simply tar'ing or zipping-up and moving all the files from the source server over to the target server and then manually trying to match everything up can be a daunting job - even for a professional. Even though tar will preserve user/group IDs perfectly, your new system will often have slightly different directory, group, and security settings that you will have to keep battling - sometimes months after the migration is complete. Instead of trying to zip up and copy over all of your content and then battle with permission problems for the next year, I would recommend:

- Creating the duplicate accounts on the new target system (either manually or via the /etc/passwd, shadow, group copy/paste method mentioned above)
- FTP downloading all content from your OLD server
- FTP uploading it back into place on the new server under the correct site and user account logins and settings

This download/upload migration method can be a bit more tedious in the initial stages, but can be much less of a hassle over time. The migration method of uploading the web site content as each of the users that owns the data will keep most of the required permissions and security settings intact--thus preventing you from having to battle the target server's OS & web expected permission and security configurations. I've seen people who used the zip-up/tar & upload method, only to end up toiling over little nit-picky file/directory/user/group permission settings for months afterwards and wishing they had just taken the time to use the FTP method.

Other apache/web issues consist of:

- Comparing the old and new httpd.conf configurations and modules of your two servers (assuming UNIX/apache)
- Verifying that system automation scripts - that may, for example, restart the web server - use the correct start/stop commands and paths
- Checking the paths of logs that are written and read (if you're doing any log analysis reporting)

One big web migration issue that is often overlooked is checking to make sure that your SSL engines (underlying SSL server programs) on your old and new servers are the same, or at least certificate-compatible. If they are not, you could be looking at either re-generating and re-purchasing all of your SSL certificates (which could take up to two weeks per certificate, and hundreds of dollars each), or installing a different SSL engine on the new server.

Page: 1 2 3 Continue >>

Server Migration Overview (Page 2 of 3)

By: Tom Weeks Corporate Technical Trainer - Rackspace Managed Hosting [view other articles by this Author]

● Not Helpful ● Partially Helpful ● Helpful ● Very Helpful Rate Article Average Rating: ★ ★ ★ ★ (Very Helpful)

E-mail Issues
E-mail is tightly bound to DNS, but there are some nice tricks that you can use to aid you in e-mail switchovers. One of these tricks is using MX records to failover your mail from the old server to the new server before the web part of the switchover.

When your new web/mail server is almost ready for you to switch over to it, the DNS TTL has been brought down to 5 minutes (more on this later), the user accounts/passwords are recreated, and you've verified that the web services and v-hosts look ok, you start your move by configuring the mail services and preparing for the DNS level change of your mail pointers.

One of the best ways to ensure no loss of mail during this process is to set up primary and secondary MX records for each domain that you're migrating. MX records are simply a DNS-based method for creating mail routing priorities.

NOTE: This is only for DNS/Internet standard (SMTP) based mail services. If you have site-to-site based MS-Exchange mail or similar, you'll need to look at your specific NT-Domain and Exchange configurations.

You set up an MX record by listing a mail server, along with its mail routing priority relative to other mail servers. The DNS entries for your old and new mail servers might look something like this (in the DNS zone file for mydomain.com):

```
mydomain.com.         22331 IN MX     10      old.mydomain.com.
mydomain.com.         22331 IN MX     20      new.mydomain.com.

www.mydomain.com      51351 IN CNAME server1.mydomain.com
old.mydomain.com.     51351 IN        CNAME   server1.mydomain.com.
server1.mydomain.com. 51690 IN A              192.169.128.55
mail.mydomain.com.    51351 IN CNAME server1.mydomain.com.

new.mydomain.com.     51351 IN CNAME          server2.mydomain.com.
server2.mydomain.com. 51690 IN A              192.169.6.12
```

This simply shows that mail should be routed to old.mydomain.com first, with new.mydomain.com as the failover server. The 10 and 20 numbers are not important in their scale, simply in their relationship to each other. 10 gets mail before 20. If 10 fails or is overloaded, the mail is then sent to the server with the MX level of 20. It would work the same way if the MX values were 1 and 2 (respectively).

Then, of course, below the MX records, the old/new names need to ultimately resolve to an IP address. So to this end, we have the MX records pointing to the old/new names, then the CNAMEs (alias pointers) pointing the old/new names to the server names, which via A record point to the actual IPs. This representation is nice because it simplifies troubleshooting if something should come up later, and IPs can be changed in one place for each server. But this also allows us to swap MX record levels to dictate mail routing priority (more on this later). Also, notice that www.mydomain.com is still pointing to the old server1.mydomain.com at this point.

Now, with this configuration in place, your mail service (at least) is ready for the move. After your TTL=5 minutes (see the DNS TTL section) and your new DNS settings are propagated out to the world, you could literally shut off the old source server, and mail would fail over to the new target server.

NOTE: This MX/fail-over setup is simply a good robust mail configuration to have, even if you're not migrating from one server to another. Most large corporations run with this type of setup so that if the primary mail server goes down, you have a backup mail server in place that catches all the mail until the primary mail server is available again.

We're still not ready to shut off the old server - we still need to move some more data and take care of the final DNS switchover steps (covered in the DNS section).

NOTE: To ensure your people are all still able to read their mail, their mail clients should be using mail.mydomain.com. Five minutes before swapping your MX levels, you change mail.mydomain.com from pointing to server1 to server2. Client mail retrieval should now be ready to go. Next, you toggle the MX 10 and 20 values, effectively telling the Internet that your new target server is the default mail server for mydomain.com.

Just remember that, before the moment of the MX priority level toggle in DNS, your clients will have to read their mail on the old server and briefly stop using mail until the switch is complete.

After these mail and DNS issues are configured and your new server is handling all of your domain mail, continue on to wrap up the rest of your migration.

Advanced Mail Switchovers
There are some more advanced methods of mail switchovers, but they involve the use of programs like rsync and procmail. You may want to look into such methods, but they are beyond the scope of this overview.

Database Migration
Database migration can be a sticky subject, and should probably only be performed by a qualified DBA. That being said, there are two ways that databases are usually migrated: either via exported flat files or by shutting down the DB, tar'ing up the whole directory structure and moving it all.

The migration method of dumping your DB to flat files and importing on them on the target server is usually the standard and orderly way to perform your DB migration, however this method can be a little more time-consuming and detailed.

Another strong-arm DB migration method that is not always recommended (unless you're running a small to medium-sized DB - MySQL or the like) is the tar & copy method. If your admin is comfortable with it, he may be able to simply shut down the DB, tar up the whole data base directory structure, and if the target server is running the EXACT same version DB, simply untar it out over the top of the target server DB dir structure (after backing up the original target server DB dir structure, of course). This can be a quick and dirty migration strategy if you take the appropriate precautions, really know what you're doing (and are willing to take higher risks).

Talk to your admin or DBA in detail about the best course of action for your situation.

Security
This too is a big category, so I'll just mention the basics of what you'll want to check out and compare on the new target server. Some of these things are:

- TCP-Wrappers or IP based ACL (access) controls
- IP-Chains/IP-Tables configurations
- FTP access rules
- Mail/SMTP relaying rules
- Mail/POP-locks and access rules
- Firewall ports, IP rules
- DNS zone transfer lock-downs/allows

<< Back Page: **1 2 3** **Continue >>**

Server Migration Overview (Page 3 of 3)

By: Tom Weeks Corporate Technical Trainer - Rackspace Managed Hosting [view other articles by this Author]

Not Helpful Partially Helpful Helpful Very Helpful Rate Article Average Rating: (Very Helpful)

DNS Issues

The TTL Issue

DNS is one of the biggest "gotchas" when moving a domain. You need to make sure that you are listed as the administrative and (optional) technical contact for the domain. This needs to be taken care of well in advance (i.e. months) of starting your move, not once it has begun. You can see who is listed by hitting a whois database:

http://whois.geektools.com/cgi-bin/proxy.cgi

If you are listed as the administrative contact, you can change the name servers for your domain name. You can also pay a provider to manage these entries for you (or run your own DNS), and ultimately control your domain and where it resolves to.

NOTE: If you're running your own DNS on the same server that you're migrating, it is not recommended that you transition both your DNS server and the domains that it resolves at the same time. It would be best to have a separate DNS server that can be moved later, or split the move into two parts over the course of a few weeks.

Even though controlling your own DNS means that you can switch your IP over yourself, there is still the issue of "propagation delays". Some providers will claim that DNS propagation will cause you a week of downtime--and if the DNS switch is not handled correctly they're right! But this is just a scare tactic to keep you from leaving. There are measures that you can take to ensure that this propagation downtime does not occur.

Most all client side ISPs hold your various domain's DNS information in a "DNS cache" for the amount of time specified in that URL's domain's zone configuration. The variable that controls this DNS caching time is called its Time to Live or TTL setting, and it tells your ISP's DNS cache to update itself (regarding your IP address) at a certain time. Usually, most DNS servers' domains (or zones) are set to a default TTL of either one day, or one week. This means that if you moved your domain, and updated your DNS, all the ISPs and caching DNS servers on the planet could be pointing to your old IP-location for a day or - potentially - a week! You can address this issue by planning ahead. First, find out what your DNS's TTL for your domain is, either by calling the people who run your DNS, or by querying the DNS server yourself. On UNIX/Linux, you would type:

```
$ dig whitehouse.gov q-type any

; <<>> DiG 9.1.0 <<>> whitehouse.gov q-type any
;; global options: printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 17386
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 3

;; QUESTION SECTION:
;whitehouse.gov. IN A
.
.
.
;; AUTHORITY SECTION:
. 10617 IN SOA A.ROOT-SERVERS.NET. nstld.verisign-grs.com. 2001111900 1800 900 604800 86400 <-----------TTL
...
```

This domain's TTL is 86,400 seconds, or 1 day. This means that if you change this server's IP address and correctly updated the DNS, all other DNS servers who have sent people to your old IP may continue to send them to the old IP for up to 24 hours. Imagine how bad this could be if your TTL was set to one week!

So what's the fix? If you control your own DNS, simply change the TTL on all of the domains affected to five minutes (300 seconds), or have whoever runs your DNS to do it for you. Then just wait the 24 hours (or your old TTL period) for all of the caches in the world to update. Now your TTL is 5 minutes (worldwide) and you're ready to switch your domain(s) over to your new IP(s).

The Final DNS Switch
Now that you have your TTL down to five minutes, finish up FTP-ing each of your web sites' data over (as the appropriate user/site), and when you're ready, simply make your domain point to the new IP address. In five minutes, the whole world will be going to your new dedicated web/mail server. The domain/zone DNS configuration should now look something like this:

```
mydomain.com.        22331  IN  MX     20  old.mydomain.com.
mydomain.com.        22331  IN  MX     10  new.mydomain.com.
```

```
old.mydomain.com.       51351  IN  CNAME  server1.mydomain.com.
server1.mydomain.com.   51690  IN  A             192.169.128.55
mail.mydomain.com.      51351  IN  CNAME  server1.mydomain.com.

www.mydomain.com.       51351  IN  CNAME  server2.mydomain.com
new.mydomain.com.       51351  IN  CNAME  server2.mydomain.com.
server2.mydomain.com.   51690  IN  A             192.169.6.12
```

After your domains are all moved over and stable, you'll want to change your domain's TTLs all back to their previous 1-day/1-week settings. This is an important final step as it prevents unusually high DNS loads (caused by five- minute TTLs) from overloading your DNS server(s) over time.

Final Suggestions

Plan, Plan, Plan!
This whole process needs to be planned out thoroughly with the sys-admins, DBAs, and web people from the start. Be sure that everyone from your sys-admins, to your content folks, to your end users know what the plan is, what the time frames are, and who the points of contact are in case something goes wrong.

Backups, Backups, Backups
You hear this a lot, and there's a reason why. Backups are like insurance. - - you'll never know how much you needed it until you actually do. To be sure that you have good backups, have the latest FULL backups (not incremental backups) sent to you and verify them for yourself. You don't want something going terribly wrong and leaving you high and dry with no backups. This is one thing that you, as the person responsible for the migration, need to attend to yourself. Do not delegate this or trust that someone has taken care of this for you.

Piecemeal Your Move
In moving your domains/server--it's usually best if you can afford to keep both your source and destination servers on line for a few weeks to a month. Just in case something unforeseen comes up, you can always switch back to your source server. Being able to move one domain at a time is always less stressful. Doing so also gives you the time to check each part of the move: proper DNS records, directory/file permissions, UID/GIDs, web/script configurations, security settings, mail settings, etc.

Summary
A full, dedicated server migration can be carried out in an orderly fashion and be completed without a hitch. The keys are to lay out all the issues covered here with your team, communicate the solutions and timeline clearly, and be sure to open the discussion to alternate concerns or other total system side effects.

Happy Hosting!

About the author:

Tom Weeks has been working with Rackspace Managed Hosting for over 2 years, functioning as corporate technical trainer, while liaising between their support, security, product development, and the R&D teams. For more information visit http://theweeks.org/toms-stuff/tweeks-bio.html

<< Back Page: **1 2 3**

Author Information:

About Rackspace Managed Hosting

Rackspace Managed Hosting provides its customers with full-service managed hosting solutions, including state-of-the-art data centers, customized servers, burstable connectivity, server software and 24 x 7 fanatical support of all hardware and core software. With Rackspace's SmoothScaling capabilities, e-businesses can add bandwidth, advanced services, or server capacity on demand. Monthly fees range from $300 to $50,000 per month, depending on the complexity required.

Founded in 1998, with locations in Texas and London, the company manages thousands of servers for customers in more than 60 countries. Rackspace was recently named The Top Dedicated Server Company in Web Server List's Hosting Awards and voted The Top Dedicated Server Host by WebHostMagazine.com. For more information, visit www.rackspace.com, or call 800-961-2888.