

**“Down on the Farm”
Network Rendering and Autodesk Backburner**

By

Gary M. Davis

Visual Effects Artist and
Certified Application Training Specialist: 3ds Max, Combustion and Toxik

//visualZ.com

Foreword:

Prior to it's creation, this document was on my mind for a long time. The reason for its creation is that for a while now, I have traveled doing consulting, production, teaching and demo work and, quite often, the issue of network rendering seems to come up. The problem is I rarely have time to discuss or cover as much as I'd like to about any topic, so I typically go over the application at hand and rarely get to go *in depth* about the non-glamorous stuff like networking for graphics production pipelines.

This is not intended to be a replacement for the user manuals and documentation to Autodesk Backburner. I want to let you, the reader, know this paper is unofficial and has no tech support. That is, it's just one dude's take on this whole matter and is not the end all, be all resource for network rendering or Backburner. However, you should not be afraid at all because the "networking" part of this is really easy, only needs to be set up once and the advantages are insane. I am not a professional network administrator, a Microsoft certified technician or an employee of Autodesk. I am, however, a computer artist that has been using 3ds Max, Combustion and Toxik since each of their v1.0 inceptions to the public. That is, most certainly, not to say I know everything there is about any of those applications, but I have tried many workflows, hardware configurations, and different approaches to working with managed network rendering since around 1992. After a while, it seems I have gotten a lot of the kinks ironed out and just want to share a few thoughts and ideas on the process. I take no responsibility for any issues incurred or problems introduced by attempting anything contained in this document. That sounds harsh, but you get the idea... take this document at face value, with a grain of salt, etc. PLEASE take the time to join the Autodesk support forums, which are a daily wealth of FREE information. <http://support.Autodesk.com> and <http://www.THE-AREA.com>.

NOTE: This paper focuses on the workflow within Autodesk 3ds Max and Combustion, but other programs such as Toxik and Flame also uses the exact same Backburner application(s) to perform their background rendering.

Introduction:

Many facilities I have visited in the past are not taking advantage of network rendering and it always kills me to see this. A typical example is an architectural firm who has something like 35 people doing AutoCAD and then one person who is "the 3D studio guy". This poor soul is often doing the most processor intensive work and rendering complex animations on a single computer. It probably takes his machine many days to render a single animation, while the other people's computers sit idle (or turned off!) at night and on weekends. For NO ADDED COST, he can configure his network to utilize 100% of the facility's processing horsepower to get done during a lunch hour do what might otherwise take an entire weekend for his one computer. Bosses like hearing this and you will undoubtedly be a hero; if you can get 35 times the amount of work done just by setting up a few software applications on computers *that you may already have in the office*.

NOTE: network administrator folks often hate dealing with artists and you may have to fist fight them to get network administrative privileges to set this all up. Be warned. Go for the knees.

A *hugely* under-emphasized fact about both 3ds Max and Combustion is the fact that they both ship, out of the box, with a FREE network rendering system. The new software since around mid-2002 for network rendering in these two applications is called "Backburner". With the purchase of ONE copy/license of Autodesk 3ds Max or Combustion software(s), users can legally network render on up to 9,999 computers (and they can each be multi-processor) using Backburner. This means you can spread the horsepower of many computers to one 3D

animation or compositing job. One machine takes charge and dishes out tasks for the rest of the computers to work on. This set up is often called a “render farm”. In a nutshell, one machine will start rendering frame #1, another will start #2, yet another will do frame #3... and so on. When the first machine is done rendering frame #1 it is given a new task (frame to render) that has not been started yet. This method of network rendering, often known as “distributed” rendering, is the way Backburner processes animations and composites. It’s a dream for mass-production of animation frames.

Other network 3d rendering systems commonly use a system known as “bucketed” rendering. With this method, several machines can optionally render *the same frame* at the same time. This is known as “distributed rendering” and is not the same as network rendering (though they both use a computer network... confused yet? No worries.). Some well-known examples of a *bucketed*, network rendering engines that are also capable of distributed rendering are Mental Ray, Brazil, V-Ray and finalRender. To be clear, Bucketed rendering is *not* the focus of this paper but is just another way of doing things. I’m not saying one is better than another... they are both merely apples and oranges way of having many computers work on rendering the same job at the same time. In short, bucketed/distributed rendering is phenomenal for test renders while you are in the process of scene creation and doing material/lighting test renders, but it doesn’t do much in the way of speed advantages to true network rendering as described herein. Bucketed rendering is really only one way to speed up rendering a single frame with multiple machines; it doesn’t make a final animation render any faster. To add to any potential confusion, one could, theoretically, do a distributed, bucketed render through Backburner - though this is really a moot point and there is no speed advantage to this because the same machines are doing the overall task at hand.

Autodesk Backburner is, in itself, three small applications that can run one or several computers on a network. All three Backburner applications are standalone programs that basically are just task management software. *None of them do the actual rendering themselves*. All Backburner does is talks to other software applications and dishes out tasks for them to do. Some graphics software applications that Backburner can control are Autodesk 3ds Max, Combustion and Toxik. I have heard that many developers around the world are working on more uses for Backburner’s control of networked computers. Since Backburner controls all rendering done by 3ds Max and Combustion, the same network rendering management software controls and monitors them both as well. This often simplifies/allows complex production pipelines without the need for custom or proprietary software tools. Setting this up is usually a one-time thing and does not require rocket science or any practice of voodoo mumbo jumbo. It’s really a snap and the benefits are phenomenal.

Advantages to Network Rendering:

Wow. Where do I begin?! There are so many advantages to network rendering that I can barely contain myself. Studios numbering from one person on up to hundreds of artists can all get the exact same benefits from using network rendering. A single computer up to thousands of networked computers can use Backburner... and all the software for this is probably already in your hands.

The Render Queue

Perhaps the biggest benefit to network rendering, aside from reducing overall project render times, is the concept of a “render queue”. The render queue allows the artist(s) to submit jobs to a computer that literally lines up jobs for network rendering. When one job finishes, the next begins automatically.

Without a queue, you are basically limited to either creating or loading a single project at a time and then hitting "render" from within the application. Then you walk away while the job renders. The problem with this is when the job is done, you have to close the project and create or load another to begin rendering the next one. This is insanity and wastes valuable processing time. *Even if you are only using a single computer*, the advantages of a render queue should become painfully obvious after a very few sessions with it.

For example, assume you send ten jobs to the queue and then go home. Backburner will manage the jobs, dishing out frames to render, and as one job finishes, Backburner starts the next task. All the computers rendering will write frames to one specified location on the network. Even if you come back to work the next day and several machines are still rendering, you can selectively stop computers from the rendering process. This means while some of the other computers are still pinned at 100% processing power, you can work on your computer creating more content, emails, and whatever else (I see you office gamers out there!) without feeling the "pain" of a render using 100% of the CPU power of your machine. As people come into the office, they can stop their systems from rendering (with a whopping two mouse clicks) and go about their everyday work. The queue remains in tact and the jobs are, in effect, paused. Then, with a double click, everyone in the office can walk away from their machines for lunch-hour or go home for the night/weekend. Their machine(s) jump right back in the queue and resume the tasks at hand. This means that instead of the computers only using an average of around 30% processing power when someone is in front of it and *none* when there is no one present, the computer now uses 100% of its power when unattended. It's at all-out full capacity. More value for your investment! It can be worth it to even enable rendering when you know you are going to be on a long phone call. You can resume the queue on that machine for the few minutes. Each frame completed by any computer you have adds up quick. Believe it.

Revisions

There is another added, hidden benefit to network rendering. Revisions. What I mean by revisions is this: because you are network rendering, you can process more versions of the same animation in the same amount of time you used to do one version. Instead of saying "now I can do in 1 hour what I used to do in five!" you might think of it as "I can get five times the amount of work done in the same amount of time as I used to!" These are two very different things.

Say you have a job and it will only take about one hour to render it. Instead of rendering one version, why not render ten or so overnight in the queue? This lets you do things like lighting studies or different versions of your project. In the morning, you can see which one turned out best and toss the others into the virtual garbage. This is an excellent way to get better as an artist. Render tons and tons of frames knowing full and well you are going to throw most of them away (but have learned from them after a minute or two of analysis). Many studios I know network render out all the tutorial files just to help them learn the application.

One popular trick often employed by 3D animators is to create, for example, a 450-frame animation but render it from two different camera angles. Now you have a total of 900 frames of animation to edit together. Quite often, depending on the subject of the animation, you can't even tell it's the same animation. This is typically because the viewer's eye is drawn to something different the second time around. Heck, why not a third, fourth or even fifth view? Now you have 75 seconds of animation to use in your edit session instead of only 15 seconds. You will have far more options to present to the client, and it requires little to no additional human intervention. The render farm takes care of it all through the render queue while you are busy with your family, being stupid at happy hour, or doing whatever it is you do prefer to do when not at the office (I probably don't want to know).

Sequential Files

Many people like to work with and render AVI and MOV files for obvious reasons. As single files containing many frames, they are easier to manage than dozens, hundreds, or even thousands of individual frames of animation. Typically, large quantities of sequentially numbered frames of animation or video require a directory for each set, as opposed to several AVI or MOV files being able to reside in the same directory and still be relatively organized. However, there are huge advantages to rendering sequential files through Backburner, *even if you are only using one computer in your studio...* Say what? Why network render on only one machine? 'More on that in a second...

One great reason to render multiple numbered still-image frames is the fact that you can distribute the workload to multiple machines (network render). By their nature, AVI and MOV files can only be created one at a time and only by one computer. You can't have six machines all creating the same MOV file at the same time. It doesn't work like that. The same holds true for AVI. Several machines can *read* the same MOV or AVI *after* it has been created, but several machines cannot *create* the same file at the same time. This is very important to understand about network rendering.

Another reason goes back to the advantage of the render queue. Hypothetically, if you had only one computer, you *could* network render AVI or MOV files and even queue them up for processing, but you would not be able to stop the render, unload the system memory (RAM), work on other projects, and later resume the render where you left off. In addition, if you were rendering and stopped (or crashed) in the middle, you would often have to start over from scratch, instead of resuming where you left off. Even if you have only one computer for rendering, it's still a fantastic idea to render sequential files to a directory for later compressing into an AVI or MOV. These finished frames will encode faster and better, and once you have the final AVI or MOV you can backup and or throw out the sequential frames used to make it.

Yet another thing to mention regarding AVI and MOV files is that, unless you are rendering uncompressed files, the compression involved with almost any codec will "do a better job" if you compress the animation *after* the individual frames are first rendered. Without going into too great detail, most compression algorithms "want" to encode a frame while looking at the previous, current, and next frames to optimize the compression. If you are rendering an AVI or MOV file and compressing *as you go*, the current frame has no "next frame" because the process hasn't created one yet. Therefore, the compression algorithm isn't optimized and your footage wont (technically) look as good. Compressing finished, sequential animation frames into an AVI or MOV *after* they have been rendered will almost always result in better looking clips regardless of the compression codec used.

Rendering sequential files will typically take the format of **filename0000.tga**, **filename0001.tga**, **filename0002.tga**, **filename0003.tga**, and so on. Whether or not you number the first frame "zero" is entirely up to you. It's common in animation to call the first frame "zero" and it is common in video editing to call the first frame "one". Again, it's however you want to work...

Frame 0-449	450 frames is exactly 15 sec at NTSC non-drop frame rate
Frame 1-450	450 frames is exactly 15 sec at NTSC non-drop frame rate
00:00:15:00	SMPTE time code is exactly 15 sec at NTSC non-drop

All three of the above examples are the same duration, but notice the differences in bold in the first two examples. Time will be expressed one frame different in your application depending on how you number the first frame. The default time in a new 3ds Max project is 0-100 (for a confusing, total duration of 101 frames). However you want to work is your call, it's just a good idea to know the difference when dealing with several people working on the same project ("dude, make sure to blow up the spaceship of frame 86, not frame 87!").

One last great feature about rendering sequential files is that you can check a render as it's happening. For example, say you have a 1200 frame animation. Around 350 frames into it, you can open up the files already rendered and check them in an application like 3ds Max's RAM player, Combustion or Iridas' FrameCycler. While the render is still going on, you can open the files that are already done and check to see how things are progressing. If there are any glaring errors, you can stop the job and the next in the queue will start, you make the proper fixes and resubmit the job. Then you didn't have to wait for the entire job to finish to see the errors. Again, time saved means happy artists, bosses, and clients.

Monitors and Logs

Another advantage to the Backburner is the fact that you can monitor your render farm from anywhere in your facility and diagnose its progress, results and shortcomings. The ability to know which machine is rendering faster than another can help you maximize your render farm's efficiency. For example, maybe you have a computer with a slow processor but with a lot of memory and a faster computer with little memory. It is advantageous to see which is rendering faster. Sometimes the memory requirements for a particular project far outweigh the processing power, and the slower processor computer might actually be rendering more frames. It would be good to know these things for obvious reasons.

Ideally, render farm machines are all clones of each other, but this is rarely the case. In the situation when they are, indeed, all identical machines, it still can be good to see speed comparisons, for if they are all identical in rendering speed except for one, perhaps there is something that needs addressed on that particular computer. In the more realistic situation where you have machines of all different configurations, you can know which ones are the beefy render machines and which are not so powerful. Believe me, it can surprise you which ones are rendering the most frames. Then when you have multiple artists rendering multiple jobs, you can easily prioritize which machines will assist in rendering which jobs, if you so chose.

Monitors and render logs are great because you can easily identify unique frame problems with a network rendering. Say for example you render 1000 frames on 10 identical computers. When you compile the frames into an AVI for editing, you notice that every so often there is a frame that has some form of anomaly such as a flickering texture map. Using the render log, you can easily identify which machine rendered those bad frames. To fix this you first need to delete the bad frames using a thumbnail viewer like Slowview, ACDSee or Thumbs Plus. Then, you resubmit the entire job to the render queue... however, this time you leave out that problem machine from the job and take advantage of the "skip existing frames" option within both 3ds Max and Combustion. Now, while the good machines render the missing (deleted) bad frames, you can go troubleshoot the problem machine. More multitasking!

The Three Backburner Applications:

As previously mentioned, there are actually three applications that make up Backburner. These are named Manager, Server and Monitor. The first two can run either as a Windows service, or as a user-run application like any program you are used to running in Windows. Setting up the two first applications as a Windows service might seem advantageous because, once configured correctly, they are running at all times. However, I personally think there are several good reasons not to do this...

From my own experience, I much rather run Manager and Server as executables and not as services of the Windows OS. This is because it is often much easier to control when a machine will render and also to diagnose any issues with a particular machine. Examples include finding a

reason it is crashing, why it might be the only computer to not be rendering certain project, or why it might be rendering bad frames. You can, as an option, merely put shortcuts to the Manager.EXE or Server.EXE files in your windows start up, so every time the machine boots, they run! You can also set up Windows to automatically log you on so that you can literally just press the power button and walk away from your machine. It will auto log on, run the appropriate Backburner applications and begin rendering ASAP. Please check the Microsoft docs or web site for information on how to do this to your Windows startup or to make Backburner run as a Windows service. The advantage to this is that the application will run at all times when the machine is booted into Windows. The down side is that it's often hard to tell at a glance if the machine is rendering.

The information below is just some of the major features of each application. As with anywhere else in this document, please consider reading the actual manuals and documentation for the technical specifics to each.

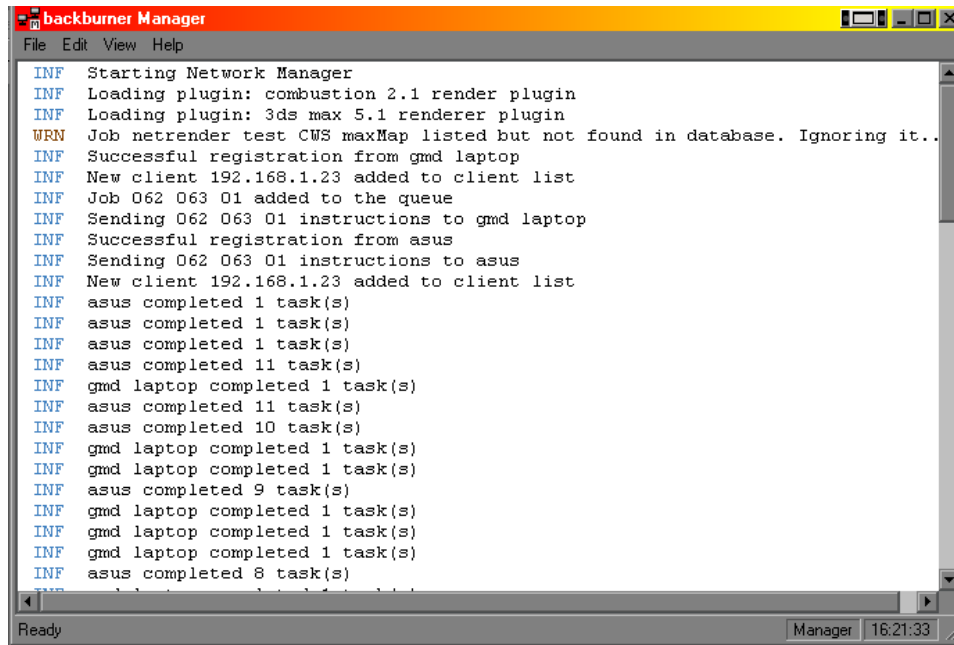
Manager.EXE – This is the grand master control of Backburner and the shepard of the farm. *This application only needs to run on a single computer and it controls the entire render farm.* Read that last sentence again. This application does not require a lot of memory to run. Remember, Backburner is only task management software. It is 3ds Max and/or Combustion that will be doing the actual rendering. The Windows service version of this application is named ManagerSVC.EXE.

In an ideal setup, the Manager machine is NOT going to be a computer that renders. This is primarily due to the fact that rendering is a very stressful process on a computer's CPU(s). If one rendering machine on the farm crashes, the others will continue to do the work and pick up the slack (even rendering the frame that machine crashed on!). However, if the *Manager* PC crashes, the whole farm stops and the queue comes to a halt. This is a very bad thing and I don't like to take that chance on deadlines.

In a smaller facility, an ideal machine to be the Manager is a spare, old machine (old laptop laying around?). Again, the files that are being rendered will potentially take up all the memory of the render farm, but the Manager application demands very little of the host computer. When upgrading office machines, I like to have one of the weaker hand-me-down computers become the Manager.

In a larger facility, it is often a good idea to have your network contain a machine running Microsoft *Server* that is nothing but a texture map vault, a Backburner Manager, and a place with plenty of empty drive space. When you have more than twenty or thirty machines, you will probably want the Manager to be a pretty decent machine to handle all this network traffic. This is to be a temporary (or permanent) destination of all the frames you will render. Windows *Server* rather than *Workstation* version of Microsoft Windows will allow you to have more than ten computers reading and writing from it at the same time.

NOTE: This entire document assumes a *single* Manager render farm. That is, there is only one computer acting as the Manager. You can have multiple Managers on the same network, but you can set that up on your own after you understand the basics of a single Manager render farm. In most cases, you will probably want only one Manager anyway.

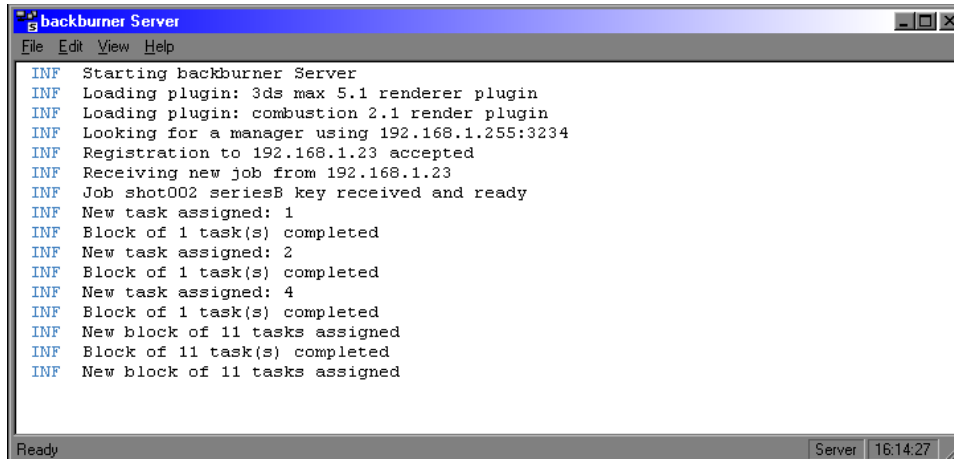


In this graphic, you can see that the Manager application is running and dishing out frames to render to either 3ds Max or Combustion. You can also see that I have two machines that immediately connected and started rendering. You typically don't really even need to watch this window, as most of the things you would find of interest here can best be looked at and edited from the Monitor application (described below) running anywhere in your facility.

Server.EXE – This application is horribly misnamed. The name implies “file server” and this is very far from anything this application does. Server.EXE is the application that needs to run *on each machine* in the farm that is going to render. This is the application that talks back to the Manager and says, “Hello there, Manager, how are you? I’m ready to kick some butt and have 3ds Max or Combustion render some frames! Do you have any thing you would like me to have them render?” Well sort of. The Windows service version of this application is named ServerSVC.EXE.

NOTE: 3ds Max and/or Combustion will also need to be installed on every computer that is going to be a render server (aka “render slave”). These installs, however, don't need to be authorized, don't require a dongle or license system, and can be very minimal installs. There is more on that coming when I discuss setting up the farm later in this paper. Both 3ds Max and Combustion have install options named something like “minimum install” which is usually ideal for a render slave machine. Manually/optionally, you can eliminate the help files, samples and tutorials for starters (a render-only machine will never need these).

If you arrive to work, a machine is rendering and someone needs that particular computer to go about his or her regular tasks, just stop Server.EXE from running on that machine. That single computer will stop rendering and you can use it for other things. However, the render queue will still be running and the rest of the rendering computers will pick up the slack. To resume the rendering on that computer, the user merely needs to launch their Server with a double click, and walk away. The user needs to know nothing about graphics, networks, video or anything else. If someone can check email, they can handle this small task. Another option is simply adding the Backburner Server application to the scheduled tasks within Windows. By doing this, you can simply have the application launch when the workstation is idle for X amount of time.

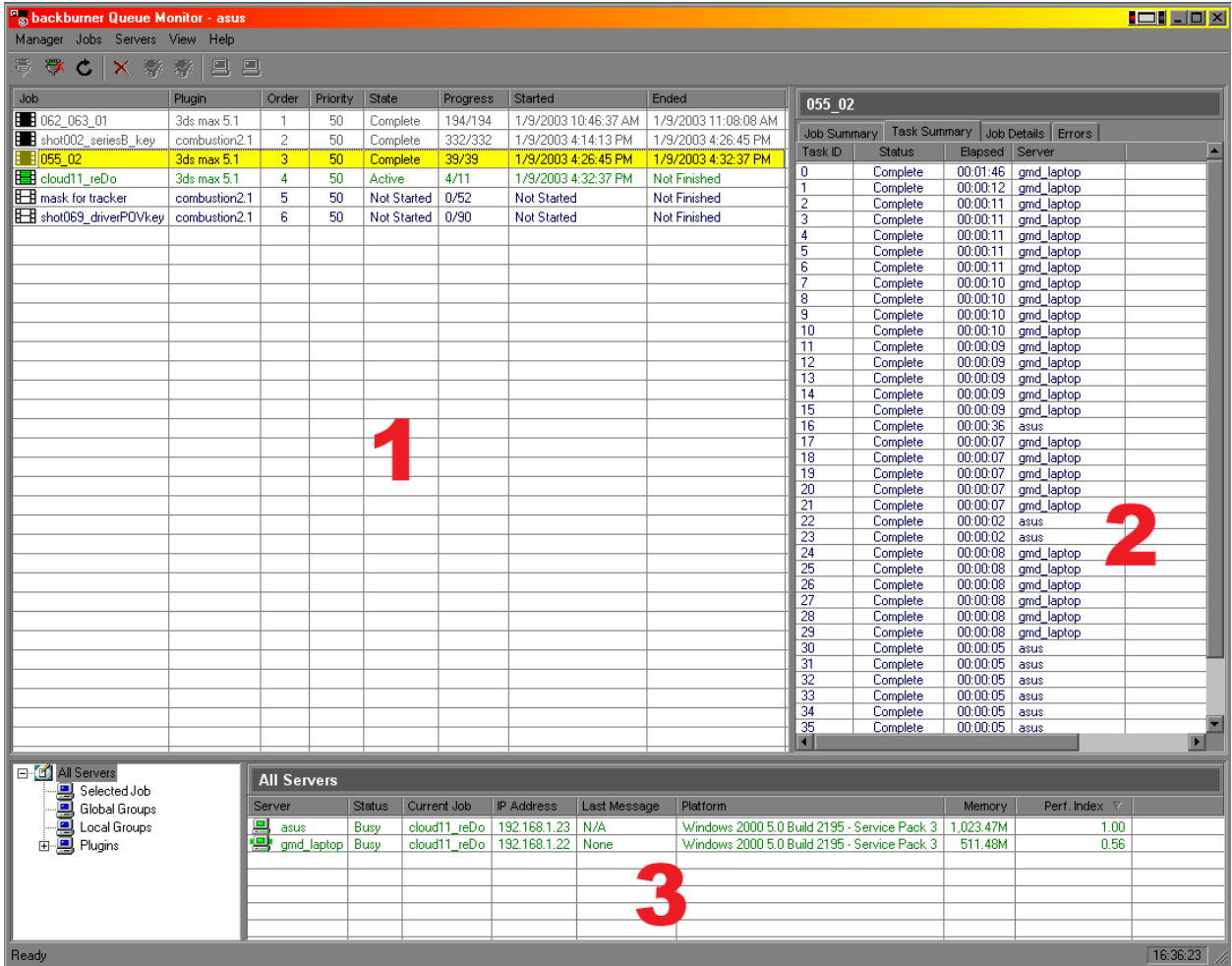


```
backburner Server
File Edit View Help
INF Starting backburner Server
INF Loading plugin: 3ds max 5.1 renderer plugin
INF Loading plugin: combustion 2.1 render plugin
INF Looking for a manager using 192.168.1.255:3234
INF Registration to 192.168.1.23 accepted
INF Receiving new job from 192.168.1.23
INF Job shot002 seriesB key received and ready
INF New task assigned: 1
INF Block of 1 task(s) completed
INF New task assigned: 2
INF Block of 1 task(s) completed
INF New task assigned: 4
INF Block of 1 task(s) completed
INF New block of 11 tasks assigned
INF Block of 11 task(s) completed
INF New block of 11 tasks assigned
Ready Server 16:14:27
```

In the above image, you can see that both 3ds Max and Combustion render engines are loaded on this machine, and that a job immediately began when the Server application was launched on this particular computer. You can also see frames are rendering without errors.

Monitor.EXE – This application has several great features. It is basically a front end or GUI to [the] Manager. However, this application can be run on any machine on the network to monitor (hence the name) the render queue. You don't necessarily have to run Monitor on the same machine as Manager. This means from anywhere in the facility, anyone can launch Monitor on their computer to see the progress of the render farm. Once connected to the Manager application, Monitor can get valuable information pertaining to all the jobs in the render queue. Examples include the speed of each machine, which jobs are done, currently rendering, and next in line and which machines rendered which frames.

NOTE: Assuming nobody on the network is running Monitor, the first time it is run will create the *controlling* Monitor. From this unique machine, the queue cannot only merely be looked at, but you can also do things like stop and start jobs, reprioritize jobs in the queue, remotely stop and start individual render machines and even delete jobs from the queue. After one person has launched Monitor and has "control" of the queue, anyone else that launches Monitor on their computer will only be able to look at this information, but not make changes to the queue. For this reason, it is often wise to have the lead artist or technician be the one running the controlling monitor.



The above image is the Monitor application interface. You can see that it is broken up into three main areas. The **first** is the actual queue in all its glory. Here you can see what jobs are rendering, the order they are going to render, etc. If you are in control of the queue on your system, you can also do things like reprioritize jobs, turn jobs on and off, and delete jobs from the queue. When you are in control of the queue, most of the commands to do these tasks are in the *right click* menus. One of the right click menus to investigate is called the Column Chooser. Here, you can determine what specific information will be viewed in area 1.

The **second** area of the UI is where you can see data about the *individual task* selected in area 1. In the above example, the job currently selected is a 3ds Max job named 055_02. Information about this job is visible in area 2. Examples of information gathered here include which machine is rendering which frames, where the frames are being saved, and how long frames are taking to render. Note that this job is already finished and the job actually rendering at the moment is named cloud11_reDo.

The **third** and last main part of the UI is where you can see information about each of the machines on the network that can potentially render. The last column (which I have placed all the way to the right) is called Performance Index. This is where you can gauge the speed of your individual systems on the render farm. A performance index of 1.0 is the fastest machine overall on your network. The rest will be a decimal value that represents a percentage of speed to the fastest. In the above example, my laptop is 56%

as fast as my workstation. As previously stated, it can be good to know which machines are doing how much work, and this is the easiest, single way to check. If two identical hardware configured machines are drastically different in speed, then you should investigate why. The right click menus allow you to access the Column Chooser in this area of the UI as well. Here, you can determine what specific information will be viewed in area 3.

Setting up a Backburner Render Farm:

Don't be afraid to network! Setting up a render farm with Backburner is easier than ever. Under most circumstances, the hardware considerations are a one-time thing and the software requirements are also very easily manageable with just a little know-how.

Considerations for Rendering Computers (aka Render Slaves)

A major, major issue to be aware of is that the hardware requirements for a render farm machine are drastically different than those of the workstations used to author the files that will actually be rendered by the farm. For their workstations, people often spend copious amounts of money on things such as twin monitors, insane OpenGL/DirectX graphics cards, huge drive arrays and DVD writers (to name a very few toys hanging off beefy workstations). These are, indeed, fantastic tools to have for creating the projects you will render and output; however, most of these things are not required for the machines that will actually render. The only real things a render box really needs of any real significance is RAM and CPU power. Below is a list of hardware components and a few notes about each.

Motherboard – Typically, any old machine will work. One possible consideration might be to get a dual processor motherboard (see CPU section just below). Other things be aware of is the number of RAM slots, the maximum amount of memory you can put on the motherboard and the configurations of RAM sticks to get to the different memory configurations. Sometimes it is cheaper to get VGA and/or networking built right on the motherboard.

Memory – This is actually the biggy. For me, it's often even more important to have plenty of RAM than it is to have crazy CPU power. The reason for this is that the moment a computer that is rendering runs out of memory, it begins to use the Windows swap file (aka "virtual memory") and the speed of the rendering on that machine takes a dramatic performance hit. Having enough memory assures you are maximizing the machine's CPU(s) clock cycles. Different types of memory like DDR and such can greatly increase the performance of your *workstations*, but I personally have never really seen the need for super fast RAM in render slave machines. Other people may tell you otherwise, however.

Be aware that some motherboards require memory to be put in "in pairs". That is, if you are going to, for example, upgrade a machine with a single stick of 128 up to 512, you might first have to exactly match the existing 128 stick of memory and then purchase a second pair of matching memory sticks. It is advantageous to put as much memory as you think you will need for the machines in the first time and then put it out of mind. I would suggest at the very least 512mb of memory for a render slave machine. Realistically however, with prices as they are now, you can put a full gigabyte of memory in for less than 200 bucks. This will be money very well spent. Personally, I would make this the highest priority for the machines that are going to be rendering and CPU power a close second.

CPU – The horsepower behind your rendering machine. Dual processor machines are able to render nearly twice the amount of a single processor computer without the need for all the physical space or other components of a second computer. Dual processor machines are, quite

often, more desirable on the render farm than they are on the workstations. This is because rendering takes 100% advantage of both processors at all times. Rarely does a human do that (when not rendering). AMD and Intel machines are both OK in my book. You might want to try and have all of them either one way or the other. In an ideal scenario, you will minimize the variables between rendering machines. I know many facilities that use Intel machines for workstations and dual AMD processors inside super cheap/fast render boxes. Both 3ds Max and Combustion take advantage of dual processor machines.

Hard Drive - A render machine really needs to only have about five gigs of hard drive space, because all that will reside on the computer's hard drive is the operating system, Backburner, a big Windows swap file, and minimal installs of 3ds Max and/or Combustion. The actual frames that are being rendered by each machine will NOT be written to these local drives and therefore require no space on them. Instead, the frames are all rendered and immediately saved to one user-specified location on the network. In addition, it makes no sense to get SCSI hard drives for render slaves, because the speed of the hard drive is almost insignificant on these machines. Think small, cheap, IDE drives. These days you can't even buy a drive smaller than ten gigs anyway. Save your money here! Only editing systems really need fast hard drives.

NOTE: Temporary, compressed copies of the projects in the queue are sent to the render servers. With Combustion files, this is rarely that important, seeing as the compressed version of the text/ASCII files are very infrequently more than a meg, but with complicated 3ds Max scenes, that can run into the hundreds of megabytes. This can become a drive space issue if you submit several very large files to the queue at the same time. While on this topic, I would also mention this in network performance and Manager configuration considerations. Although it is not necessary to have a lot of RAM on the Manager, these compressed versions of max and Combustion jobs, do need to be sent out from the Manager, and when dealing with, let's say, a complex building with radiosity meshing enabled, this can be a task that involves a lot of memory and requires fairly good performance to avoid timing out. Admittedly, this is the exception and not the rule as far as job file sizes are concerned.

Network Adapter – a cheap network card is all you need for a render machine. Every computer will need to have a network card installed. These are called “combo” cards and can be purchased at just about any store very inexpensively. I suggest PCI cards over the older ISA cards. Quite often, motherboards come with built in networking, but I prefer to use cheap, add on PCI cards, because they can easily be replaced should you ever have any issues such as a voltage surge, etc. Combo cards will often be labeled “10/100”. This means that the hub or switch that they are plug into will determine the speed of the network, but this card is capable of doing either 10 or 100 megabits per second of data transfer across the network. Because the render machines will only occasionally write a single frame across the network, it is not necessary to have super fast (gigabit) networking on your farm unless you have literally hundreds of computers running at the same time. A few dozen computers will not require anything faster than 100 megabit networking. Older, 10 megabit networking will, however, slow you down significantly and you should really avoid connecting these cards to a 10-megabit hub or switch. You will want to, ideally, be at 100m/bit on your rendering machines.

Graphics Adaptor – This should be the big money saver. On a network render machine, you only need to see windows boot and be able to launch Server.EXE. You will never open 3ds Max or Combustion to do work on these machines. Since you will never see their interfaces, there is absolutely NO point in spending any money on a graphics card. The cheapest, nastiest cards are the way to go. A default Windows install desktop will run 800x600 and in 16 colors without any 3rd party drivers. This is often just fine; as you will rarely, if ever, look at graphics files on these machines. You should use your high-end computer workstations with expensive video cards to look at and check frames from across the network.

NOTE: for Combustion users that are familiar with the integrated, OpenGL particles system, it is still suggested to network render with software and not OpenGL... unless all your graphics cards are identical on every machine. For 3ds Max it currently doesn't matter what cards are in any rendering machines.

Cases – Since the machines that are rendering will probably only have a network card and possible a VGA card, they will typically not generate too much heat or require too much power. Workstations and file servers commonly have 300 or 400-watt power supplies, but often a rendering machine can get by with 250-watt (or even below that) power supplies. Rack mount cases are also a great consideration because you can stack several computers on top of each other in the space of one “normal” computer tower. Rack mount cases are measured in height units simply known as a “U”. For example, a 4U rack is four times as tall as a 1U case. Typically the smaller thinner/shorter the case, the more expensive it is. Typically 1U cases also require special motherboards and expansion cards. For this reason, it is often more advantageous to go with 2U cases. They are still very small, usually require no special internal parts, and sit at a nice price break point against the 1U cases. There are also new PC's now popping up that are basically little cubes. These cubes might also serve you well as stackable render farm machines, but FYI - they are typically single CPU only.

Keyboard, Monitor, Mouse (K/M/M)– Again, these are technically not even required on a machine that's rendering. Some computers need to “see” a keyboard to boot correctly. You will not even need to see the computer's interfaces when they are rendering, since you can monitor their progress from anywhere else in the building!. Think cheap here. A typical set up will have a K/M/M switcher on several machines. That is, there is only one physical keyboard, monitor, and mouse for the entire render farm, and a series of buttons enable these for one computer at the a time. Basically just when you need to log on, defrag every so often, etc. but not even every time you do a rendering job.

Other –Floppy drives and CD-Rom drives are technically not required for a network render slave, however, the price of these items is so small that it is suggested to put them on anyway. This will, obviously, help you install software and troubleshoot the system. Think cheap, slow, cheap. Furthermore, you do not need any DVD or audio capabilities for a rendering machine. SCSI controllers or devices are, likewise, not needed for a rendering machine and are really only a waste of money.

Other Networking Hardware Required

All you really need to link the entire network of computers together are cables and either a network “hub” or a network “switch”. Either of these can determine the speed of your network and also how many computers you will be able to network. I would highly recommend a “100 megabit” speed of either, as “10 megabit” speed will probably cause “traffic” and “collisions” on your network. There is a significant hit in performance when using a hub as opposed to a switch. Here is an analogy: a hub is like the old time 'party-line' phones, where all the users share a single connection, whereas a full duplex switch is more like phones as we know them today. This is because a switch can send simultaneous information to all ports as well as receive information, whereas a hub can only deal with one set of information at a time. As with the previous note, this is rarely a consideration when dealing with fairly small jobs and frames, but when more information needs to get across the network, this might become something to consider. Again, you should read into this if you are in doubt of which is better for you. I have used both hubs and switches with no problems at all.

Regardless if you get a switch or a hub, they are easy to set up. You typically just plug them in. No software required. These are quite often in a rack mount format, which are ideal for render farm set-ups. A typical switch/hub will have 5, 10, 12, 16, 18, or 24 ports on it. What's nice about these devices is they usually can also be “daisy chained” for expansion purposes. This means

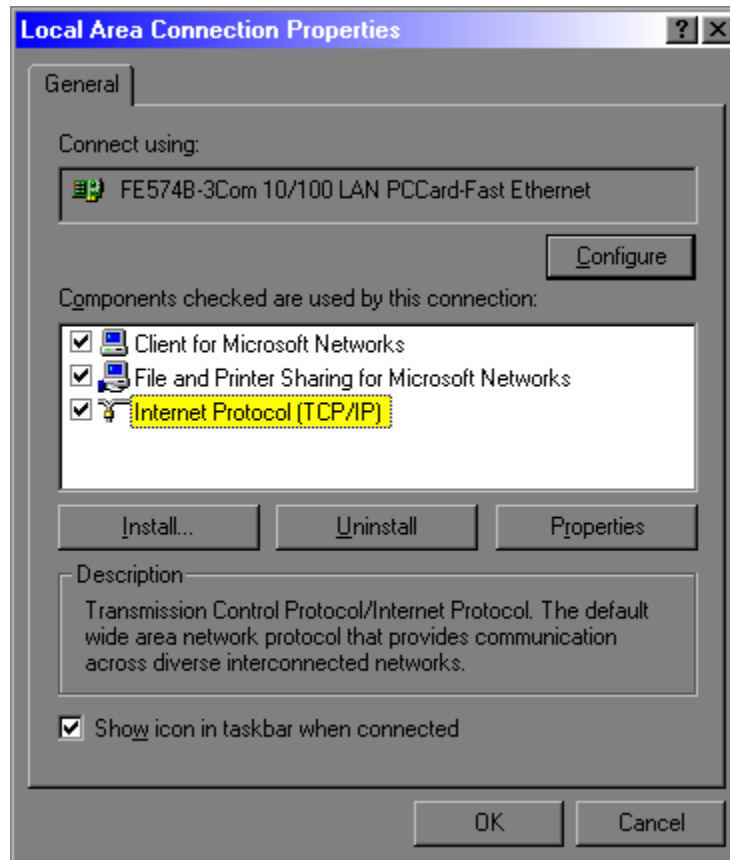
you can have several connected together, acting as one (kinda like a render farm). For example, say you have a 5-port hub and fill it with a 5 PC network. Later, you want to add 4 more rendering machines because you got a great gig but need some extra horsepower. You don't necessarily have to replace your existing hub. Instead, you can take the hub and put a single network cable between it and a new hub. All the cables running to your rendering machines can stay exactly where they are except one, which will have to go to the new hub. I would recommend getting a hub or switch with room to grow, but it's not a big deal to run out of room on the one(s) you have now. It's not a throwaway situation if you fill it.

Lastly, network cables will need to be run from each computer to the switch or hub. In older networks, you had to have one cable go to the next computer and complete a closed loop. Any break in the loop could cause the whole network to go down. Now, you just connect each computer to a central location and the networks are much more stable, as a whole.

Gigabit or fiber networks are gaining popularity because they can move massive amounts of data across a network very rapidly. Remember, however, that each machine rendering will render a single frame and then briefly write that finished frame to your destination directory across the network. In the case of video resolution, this is only around 1.5 megabytes or less at a time. Because of this, you don't need to have ultra-fast networking between all the dedicated render slave machines. Between the file server, your workstations and editing systems might be another matter, depending on your needs.

Unique Machine ID's

The biggest *software* concern when setting up a render farm is what is known as TCP/IP networking. All this means is that each machine on the network will have a unique ID and/or machine name. IP "addresses" are often better than machine names because they are very exact, but people often merely use machine names to configure their farm with no problems whatsoever. This is often only done (once) when you initially build the computer and is typically not something you just go around changing on a whim.



An important, undocumented thing to know is that machine names in a render farm cannot begin with a number. For example, computers on the farm named 001, 002 and so on will cause errors and flat out won't work. I suggest adding a character in front, but numbering them accordingly. Many people like to get cute and name their machines funny names, but I highly recommend naming a render farm with numeric names so you can easily identify them when you have to go and reboot them, for example. Names like RN_001, RN_002 are typical (where RN stands for render node). This way, if you have many computers on the render farm, the machines dedicated to rendering are all listed in the same location and in alphabetical order. It's OK to go nuts and name your workstations Frodo, Jasmine, Harley, Vulcan or whatever else, but I would recommend naming all of your dedicated render slaves numerically with a letter prefix. Put stickers with machine names on the computers for easy identification. That, or a tiled desktop image of the machine name on each computer can really quickly help identify different PCs connected to a K/M/M switcher.

If you are going to set up a subnet mask, I recommend setting the value to **255.255.255.0** as the default startup. This will narrow an internal network to search a much more narrow range of IP addresses when it "looks around" your network.

There is a unique IP address for closed, private networks. This is **192.168.1.XXX** where XXX is a number from 0-255. This lets you set up a render farm of up to 256 machines with very little effort whatsoever. It is often perfectly acceptable to have automatic IP assignment by Windows and to merely just use machine names for your render farm. This is a one-time set-up and, unfortunately, another time you may have to bug the IT folks and beg them to cooperate with you (...or go for the throat this second time around).

Networks with More than Ten Render Nodes

When you are fortunate enough to have ten or more computers rendering on your network, there is a special consideration you need to address (this is a good problem to have!). Because of the limitations of Microsoft Windows Workstation, no more than ten computers can be connected to another computer running Workstation. For example, if I have twenty computers rendering frames to an editing system, only ten of the computers will be successful in writing files to that machine *if it is running a Workstation version of Windows*. To have more than ten computers write frames to a single location, you will need the computer to be running a "File Server" version of Microsoft Windows. In all actuality, there is very little difference between Workstation and Server versions of the Windows operating systems. Price is the main one and networking/security is the second. I have a close colleague who runs Windows Server on his non-linear editing system just so he can have all the machines rendering directly to his editing system. This means no bulk/mass moving across the network of hundreds or thousands of frames in the morning.

Workflow:

Quick, One Time Set-Up

1. Verify that you have a network card in every computer.
2. Install Windows with TCP/IP networking options on every machine and set up a large, fixed size swap disk.
3. Optionally schedule a defragmentation every week or so on every machine.
4. Turn off any screen savers or merely use "blank screen" saver. These use CPU resources you don't want to waste (do NOT use the OpenGL pipes or flowerbox screensavers!!!).
5. Cable all computers to a network hub or switch.
6. Install 3ds Max and/or Combustion on every machine that is going to render. The workstations can have (up to) full installs but the dedicated render machines only need minimal installs. When this install occurs, Backburner will also be installed.
7. Decide where (on which computer and which large hard drive) texture maps and rendered frames should go. It is easiest to just set up a texture map location on the same machine as where you will render your frames. Share this location by right clicking on the directory in Explorer and select sharing. Name this share something like "netrender", "storage" or "frame vault". If there are more than ten machines, this location should be on a PC running the Server version of the Windows operating system. You might also use two different hard drives on the same machine. Consider making this a non-rendering machine and also the Backburner Manager.
8. Using Explorer's Tools > Map Network Drive... map a common drive letter to this location on every machine. Make sure you use the option to reconnect at logon. A suggestion might be to use drive letter Z:\ so that it is always listed last on every machine. This will also insure every machine on your network is connected to the same location for bitmap resources and as a location to save rendered images. I.e.:

```
X:\maps\city\  
X:\maps\wood\  
and/or...  
Z:\renders\client01\Monday\  
Z:\renders\client04\revision02\  

```

NOTE: If you want to use UNC naming conventions instead of drive letters, it is typically fine to do so. Just make sure you are totally comfortable with this workflow and remember to use UNC conventions *at all times* (for footage, textures, render locations, etc.). Please refer to the Windows help file for information on using UNC naming instead of mapped drive letters.

Daily Workflow

1. Run Manager.EXE on one machine and leave it running. This will ensure the render queue will be available at all times. Every so often, you can stop the queue and reboot the machine just to stay “so fresh and so clean”.
2. Run Server.EXE on any dedicated rendering machines and leave it running. This will allow them to wait for work at all times. You might consider putting a shortcut to Server.EXE in the startup of Windows so it runs automatically.
3. Make sure the same fonts and plugins are local (or available to) each system.
4. Build scenes and projects in 3ds Max and/or Combustion as you normally would. Make sure any resources like source video or texture maps are obtained across the network from drives and shares that all computers can see.
5. Select an output type of sequential images. You do not have to add the numbers, because Backburner will number the finished frames automatically.
6. When you go to render, set the output path to a shared network drive letter (ie: Z:\) and enable the network render option from the app’s render dialog.
7. Assign all machines to the render that you want to have working on that particular job. Typically, you can just check the “use all servers” option and all machines running Server.EXE will render. Remember, you can go to any machine currently rendering and stop /start it by turning Server.EXE on and off locally to that machine. The queue will remain in tact.
8. Submit the job to the queue and save your project. The machines running Server.EXE will start to render frames.
9. Keep working from step #4 and when you are finally ready to render, exit all applications on the workstation and the run Server.EXE on it as well. It will jump right in the mix.
10. Go home or to bed, already. Sheesh.

While Net-Rendering

Any of the following can be performed at any time while a network render is going on...

- Run Monitor.EXE from any machine to view the render queue. Remember that the first machine to run Monitor will be in control of the queue and any other people that run Monitor will only be able to view the queue.
- Spot-check any finished frame(s) from anywhere on the network in an application like Photoshop, ACDSee, or any other application. If you see any glaring errors, pause or delete that job from the queue using the controlling monitor. The next job in the queue will begin; meanwhile, you can trouble shoot the job.
- Compile any finished frames in a program like Premiere, FrameCycler or Avid to spot check animation timing. You don’t have to wait to finish a job to do a spot check.
- Build more render boxes, install 3ds Max and/or Combustion, slap them on the network and add them to the queue. They will jump right in the current job and begin rendering.

Case Example of the Whole Shabang in Action:

Here is a drawn out, hypothetical situation of one dude using 3ds Max, Combustion and later network rendering with Backburner. This is just one scenario and im not necessarily saying this is exactly what to do. Any specific vendors or products are just a few of the many options out there, obviously.

Our hypothetical dude is just one freelance artist working out of a small office. After going to SigGraph and seeing some killer demos, he purchases one copy of Autodesk Combustion and one copy of 3ds Max for around \$4K total. This initial purchase also includes Backburner because it is included with 3ds Max and Combustion. He's now ready to embark on building his multi-media empire. First and foremost, we need a workstation on which to run these fantastic applications. On this initial machine, he spends a little more than he might like to, and wisely makes sure to get a great video card and a few big monitors. Here's the computer he has built for himself:

- Single CPU motherboard with Intel P4 CPU
- 1 gig of memory
- Nice, big case with room for expansion (400 watt power supply)
- nVidia Quadro 900 XGL
- (2) 19" monitors
- DVD reader/writer
- 80 gig IDE drive
- Soundcard and speakers
- 10/100-combo network card for in-home DSL hook up to Internet
- Keyboard, Floppy and Mouse with scroll wheel
- Windows 2000 Workstation

He names the system "Hero". With this single Hero system, he can create 2D and 3D animations and even burn DVDs for both his client output and system backups. Wow... all that for well under \$4K. Not too bad considering that he's now doing work and paying the bills with one computer and a few software applications. He's in for around nine thousand bucks after a few knickknacks like Photoshop, and Microsoft Office. He is ecstatic and even more so after he ends up paying this all off after just a few production jobs for some local architects and ad agencies.

This Hero machine is his only computer, so he runs Manager.EXE on it as a Windows Startup. This means at any time he can be working in Combustion or 3ds Max and submit jobs to the render queue. When he is ready to begin rendering, he just closes all applications, double clicks an icon on the desktop, and walks away. This icon is a shortcut to Server.EXE that he has created for easy and quick access to beginning the render process. Backburner's Manager then "talks" to the server application, which, in turn, tells either Combustion or 3ds Max to begin rendering. When one job in the queue is done, the next begins. Our artist here can come and go into his office, and merely close Server.EXE when he needs to do more work. When he is ready to render again, another double click, turn the computer display monitors off (option) and go home for the night.

After doing more work for a while on his single standalone machine, he realizes he should be network rendering on multiple computers to get things done faster. Now he has to begin his dive into the creation of his first render farm. First, he purchases a cheap, 5-port auto sensing 10/100-network hub and a few network cables for around 150 bucks total. But what about the actual computer(s)? Since it's just him, he knows that any additional computer will be dedicated just for rendering, so he configures it as such:

- Dual AMD Athlon motherboard with two Athlon CPUs
- 1 gig of memory
- 2U rack case with 200-watt power supply
- 10 gig IDE drive (smallest available!)
- 10/100 PCI combo network card
- Cheapest, generic PCI graphics card available
- Cheapest IDE CDROM drive available
- Floppy drive
- Windows 2000 Workstation

Lets say this machine can be built for around \$900.00 (probably can do it even cheaper). This is a far cry from the \$4K he spent initially on his main workstation. By adding even just one of these machines, he's getting three times the render power at night (a dual Athlon plus the single P4 workstation he already has). While he's working during the day, he will still have a dedicated machine jut for rendering. After doing the math, our artist decides to get two of these machines for around \$1,800.00

To monitor the render farm, our lone artist gets a quality, 8-machine keyboard/monitor/mouse (K/M/M) switcher. This last item allows our artist hook up both of his new rack mounted render nodes (named RN01 and RN02) to a single VGA monitor, keyboard and mouse. At a garage sale, he manages to score a junk, used 15" monitor, old keyboard, mouse and even scrounges up an empty 12U rack for holding the farm (even though he only needs 4U at the moment). K/M/M extender cables from all the render slaves to the K/M/M switcher are the finishing touches. He puts the K/M/M switcher and 15" monitor on top of the rack mounted render farm and the whole thing takes up about as much space as a chair. Consider his new situation... he now has around six times the horsepower he did initially and for only an added cost of under \$2K. Remember, no additional copies of 3d max, Backburner or Combustion were needed. Think about that a sec... five times the power for less than half the hardware cost... and no additional software costs except the OS on the render slaves. In addition, these new toys only take up a small space in the corner. Sweetness.

One day, our artist is rendering a job and one of his repeat clients calls up. "We have an emergency and the deadline we gave you has to be moved up to Monday!" Their job is already in the queue and rendering, but the Monitor says it won't be done until Tuesday afternoon. Our smart thinking artist runs out (while the farm is rendering) and picks up two more of the identical render boxes for another \$1,800.00. While the render is going on, he put them in the rack, cables them up, installs Combustion and 3ds Max, and then runs Server.EXE on these two new machines. A quick look at Monitor says there are two new machines (already named RN03 and RN04) now available, so our artists add them to the job already rendering and they jump right in the battle. Monday delivery no problemo, and now our render farm is now around ten times the speed of our single "Hero" computer. To get this 10x speed in horsepower cost our artist around \$4K, or roughly the cost of his single, first, decked-out workstation.

Man, things are looking good for our dude. He's now rendering oodles of animation and getting so much better because of it. What used to take overnight he now does during lunch. He can render tons of versions, examine them, and throw away tons of frames after learning what needs tweaked. He starts to get more clients and bigger accounts.

After around 2 years, our artists realizes his Hero machine is starting to show it's age, and he begins to eyeball the new triple Intel P7 Zoltar motherboards. Its finally time for a new machine. To this brand new workstation, he adds a hardware non-linear editing card, two 24" plasma screen monitors, gets crazy ram and so on and so forth. He names his new creation "Hog". The initial Hero machine is now made into the Backburner Manager machine. Our artist gets a few hundred bucks on E-Bay for now-slightly dated NVidia card and 19" monitors. With the E-bay cash he puts in a cheap-o VGA card and a few 200 gig IDE drives. With these now connected to

Hero, he has a dedicated network Manager that can also pull triple duty as a texture map server and location for the entire render farm to spit out the frames. Knowing that he may still grow the farm, our artist thinks ahead and formats the Hero machine with a File Server version of the new Windows YQ. This is so more than ten computers will be able to render frames here if he ever decides to add more rendering slaves.

Now he has the new Hog for creating project content, a dedicated eight-CPU rack-mounted render farm and "Hero" is his network Manager/Monitor/backup machine. Ok, maybe sometimes he still uses this old pal for word processing or texture map creation... you got me there. The point is that Hero is still a viable part of this studio and has just been reassigned duties. Maybe a DLT or some form of large backup is ultimately added to Hero. All the machines render their frames to Hero's large, cheap IDE drives, and then the new Hog machine pulls these frames to its fast drive array for compiling and non-linear editing. As tempting as it may be, our artist does NOT render on Hero, however. This machine is now a dedicated resource server and Manager, so keeping the stresses of rendering off of it will be a nice break for this machine... especially after such a long service in the trenches of war... I mean production. The farm must not go down (!) but Hero can still multitask on less stressful things such as the Internet and doing archival backups while it manages and monitors the farm. Ok, ok... he sometimes renders even on his old pal and Manager, Hero. We all know about deadlines.

Our artist keeps upgrading and machines keep getting downgraded, added to the farm, etc. They do indeed, by the way, all live happily ever after, without any crashes, missed deadlines, or power surges. They lived happily ever after.

The end.

Conclusion:

Computers get faster and cheaper everyday. Even while this is the case, we, as CGI animators, still have to wait for renders. No realtime, caustic, global illuminated scenes with pyroclastic hypervoxels that have raytraced reflections... yet. Waiting for renders seems to be the bane of our existence and probably will for a few years more. The good news is that the days of the need for Irix SGI workstations and the like are extremely numbered. With the advent of insane video cards and gigs of ram for mere hundred of dollars, you can get huge 3D and 2D renders done for a few thousand dollars. Heck, many laptop computers have more horsepower than workstations from a few years ago and even out-power some workstations today! Personal computers bought at Staples or Office Depot are now workstation-class machines yet, no matter how much horsepower you throw at a graphics job, it never seems to be enough. Several colleagues of mine and I often joke that each time we double our render horsepower, its not going to take half the time, but instead, take just as long but look twice as good. We'll all just keep throwing more at the farm. Distributed network rendering is, for now, the obvious way to get a ton of work done quickly and cheaply and Backburner allows this to happen with little to no headaches. They say time is money and I'm all about saving both.

...now get to the herd and render.

Appendix A – Individual, Sequential Image File Formats:

The file formats below are *just a few* of the more common image file formats used for network rendering. There are, certainly, more file formats such as DPX, IFF, PICT, TIF, SGI and others, but I'm going to only list a few common to many facilities I have visited and worked with.

Format	PROS	CONS
<p>*.TGA</p> <p>(aka Targa) Old faithful. The Fender Stratocaster or Techniques1200 of the still images. Truevision broke the mold when they made the Targa format.</p>	<p>This format can most likely be read by any graphics application you will come across. It is very cross platform. Targa files can have alpha channels. They can optionally be compressed.</p>	<p>You will need plenty of hard drive space If you use Targa files, especially if you use the feature know as "Render Elements" in 3ds Max.</p>
<p>*.JPG</p> <p>(aka JPEG or "jay' peg")</p>	<p>They can be very small because they can be compressed. The user controls the quality of the compression. Nearly every application out there can read them and they are widely cross platform</p>	<p>The compression algorithms used by JPGs, even at 100% quality, is technically considered a "lossy" compression scheme. JPG cannot contain alpha channels. If you are going do any chroma keying in your project, avoid JPG files on the chroma key footage.</p>
<p>*.PNG</p> <p>"Ping" (aka "The new kid on the block"). This format was created initially as an alternative format for use on the web, but I have been using it recently for almost all my network rendering. I have personally never experienced a problem using 3ds Max and Combustion exchanging PNG files between each application.</p>	<p>These files are compressed in file size, but they use a <i>lossless</i> compression algorithm. Quite often they are much smaller than JPG files yet lose <i>no</i> quality! They can optionally be up to 16bits of color per pixel (aka 48-bit color space) for when you might be working for very hi-resolution output to film scanners. If you are working for video, you only need 8 bits of color per channel.</p>	<p>Not all applications can read and/or write PNG files. It's a good idea to check your editing system and any film recording facility if you plan on using these files. Make sure your favorite applications can read these, and you're in there.</p>
<p>*.CIN and *.DPX</p>	<p>Widely recognized as the formats for the motion picture industry. Analog film scanners typically output these file formats. These formats support 10 bits of color per pixel (30 bit color space) and DPX files optionally support time code as metadata.</p>	<p>These files cannot contain an alpha channel. Cineon files are often difficult to deal with due to a color conversion known as Look Up Tables (or "LUTs"). Most NTSC displays and VGA monitors can't show this much color without applying some form of LUT. Dealing with logarithmic and linear LUTs is a black art and shall not be discussed here.</p>
<p>*.RLA and *.RPF</p> <p>Autodesk 3ds Max and Combustion currently share a special link, because they are the only two applications that can <i>truly</i> take advantage of every channel contained in an RPF file. Please consult your documentation for information regarding these files.</p>	<p>In addition to the typical RGBA information contained in an image, these files can contain metadata such as Z-Depth, pixel velocity, material and object effects channels, to name a few. These images can be much higher bit depth than the typical 8 bits of color per channel.</p>	<p>These individual files are often extremely large. The use of RPF and RLA files often requires a deeper understanding of both 3ds Max and Combustion to fully take advantage of the extra data. A strong coordination between the 3D animator and the compositor is often needed (but when that's the same person, no problem!). RLA files from different 3D applications behave differently.</p>
<p>*.EXR</p> <p>ILM invented this format and it is the up and comer, for sure. This format allows optional workflow in floating point color space and also allows for extra channels of information to be stored in each frame as metadata.</p>	<p>Supports alpha channels and also up to floating point color space. Also supports extra channels such as Z-Depth, among numerous others.</p>	<p>Not all applications currently support working in floating point color space and/or working with all the potential extra channels in an EXR file. Please consult the documentation of your compositing application to determine the scope of EXR compatibility.</p>

Appendix B - Single File (Movie) Formats:

The below file types are just a few file formats that exist today that are a single file on your hard drive, yet they can contain hundreds or even thousands of frames of video or animation. These files are often easier to manage on your system because a single file represents so much information, but quite often they are actually slower than individual, sequential files when used in animation programs such as 3ds Max and Combustion. This is because unlike sequential files, these files need to be “read into” on every frame if you are, for example, using them as a texture map or as a layer in a composite. This is not to say you should never use them as texture maps, etc... but be aware they are often slowing you down to do so. This is quite often a negligible or even indistinguishable difference, however. Do some tests. Unlike animation software, most non-linear editing systems will, however, work faster with these files *if* they are in the native codec for that particular editing system.

Format	PROS	CONS
<p>*.AVI Audio Video Interleaf (aka Interleave or “Video for Windows”).</p>	<p>Many different flavors or “codecs” for different output applications from CDROM playback, web delivery and broadcast. They can also be uncompressed. DivX is a new AVI codec that encodes very fast, creates small file size, and look real nice. DivX is not intended for broadcast but is great for web, emails, and client reviews. www.divx.com. NOTE: this needs to be installed on every machine for files to read. I suggest Cinepak AVI files for playback on any PC worldwide.</p>	<p>Can only be created by one machine at a time. Cannot have Alpha channels. PC only (however, I’ve heard some Macintosh applications can now read and even write AVI files. Im not exactly sure about this. Not ideal for network rendering. Must make sure every machine has the <i>same version</i> of different types of codecs installed to access these files.</p>
<p>*.MOV Quicktime</p>	<p>Widely used in the video production field. Cross platform. Many different codecs for different output applications from CDROM playback, web delivery and broadcast. They can also be uncompressed and optionally contain an alpha channel. This is confusingly named “Millions of Colors +” (note the + character that indicates alpha channels). Many nonlinear editing systems can output a software codec MOV file that can be ready by any computer for postproduction and FX work. One example is the Avid Meridian Uncompressed format of MOV.</p>	<p>Can only be created by one machine at a time. Not ideal for network rendering. Must make sure every machine has the <i>same version</i> of different types of QuickTime installed.</p>
<p>*.MPG aka MPEG (em’-peg)</p>	<p>Highly compressed files that look very good. There are several basic types of MPEG files. “Type II” MPEG files can be broadcast quality and are 720 x 480. This is what you need to burn DVDs. Autodesk Cleaner is great software for creating these files. MPEG type I files are half as small in each pixel dimension (for a quarter total image size smaller) and are good for client review and web purposed. MPEG 4 is gaining popularity fast.</p>	<p>You typically don’t edit MPEG files. They are often highly compressed and should be considered the “end result” of your projects. You currently can’t render MPEG files out of Combustion or 3ds Max and you should not want to anyway. Again, they can look good playing back, but are typically very compressed files.</p>

FILE FORMAT NOTES - If you are ever in doubt about what format to use or if another facility can read them into their pipeline, I suggest using uncompressed TGA (Targa) files. They are the “Fender Stratocaster” of the animation file formats. That is; they haven’t changed in years and still rock the house no matter what project you are working on.

I typically work with 16bit PNG files back and forth between 3ds Max and Combustion and half float EXR files back and forth between 3ds Max and Toxik. Remember, your workflow may be totally different and work like aces for you. Different hardware editing systems will allow different file types to play in real-time, but you should still consider network rendering individual frames and THEN compiling and compressing them down to your editing system codec.

For further information on file formats and LUTs, I highly recommend Ron Brinkmann’s book titled, The Art and Science of Digital Compositing, from Morgan Kaufman Publishing.

Another is a shameless plug for my own book, The Focal Easy Guide to Combustion, from Focal Press publishers.

Tech...

This paper was originally, slowly written and rewritten in Microsoft Word 2000 on my Dell Inspiron laptop during Fall/Winter/Spring of 2002-2003. It was largely typed in several hotels, airports and in the air between Boston, Chicago, Denver, Las Vegas, Montreal, Orlando, Portland and in an Avid bay waiting for 22,873 Targa frames to export (hated it). Screen captures were done at my pad using SnagIt! on my homemade Intel P4 box running Windows 2000 (sp3). Autodesk 3ds Max 5.1, Combustion 2.1, and Backburner 2 were the applications and versions used for the main, graphics content herein.

Slight revisions for SigGraph were done in July 2005.

Further revisions for Autodesk University were done in October 2006.

A Few Thanks...

I would like to first and foremost thank everyone at Autodesk for providing the tools I use everyday to make a living. The meaning of "work" translates to "play" because of all of you. Special thanks goes to DJ Rahming and Yann Bertaud (at Autodesk) for their insight and help regarding networking and Backburner. A very huge thanks goes to the entire Autodesk tradeshow posse. um...woah. Special gracias to Curtis Sponsler of The Animill in Orlando, Florida for letting me bastardize, destroy, rebuild, mess up and fix his render farm during the tedious and fevered production of a television series and then some. We learned a lot and I hope I (we) have saved many people from the headaches we endured. Huge thanks will always go out to Ron Coleman of BGSU for instilling in me the insane desire to create art of all kinds with electronics.

About Me...

My primary education was in graphic design, math, photography, painting and sculpture. I obtained a BFA in Computer Animation from Bowling Green State University in 1992. My professional career began performing live visual FX shows while touring with many electronic recording artists for almost three years. After that, I spent around five years developing hardware systems and animations for the simulator ride-film and amusement park markets. Since 1999, I have been on my own doing production broadcast graphics, visual FX, consulting and teaching as a certified training specialist for 3ds Max, Combustion and Toxik. I currently do laundry and the majority of my network rendering in Orlando, Florida but I seem to spend a lot of time totally lost in hotels and airports. I am also a regular instructor at www.fxphd.com as well as at Orlando's Planet Digital training center. Please feel free to drop me a line about this document or inquire about my consulting, production and/or training services thru my website (below).

Any brand or product names are the property of their respective owners. This document is not affiliated in any way or official manner with any of these companies and is provided as is. There are no guarantees or tech support provided. Please read the manuals of the products you own. Please support this great industry by paying for your software. The contents of this document are freely distributable and may be printed for personal use, but cannot be reproduced for production in any manner, whole or in part, by any means (electronic or otherwise) without express, written permission of the author. You may link to this document but please mention my website if you do so.

"Down on the Farm"

© 2003, 2005, 2006 by Gary M. Davis.
No part of this documentation can be reproduced
without the express, written permission of the author.

Gary M. Davis
Application Training Specialist
3ds Max | Combustion | Toxik
<http://www.visualZ.com>