

10 *Configuring a Large Network*

If you have a large network, the basic configuration supplied by SimpleNetworkStarter probably won't be sufficient to meet your needs. Two features of large networks call for additional configuration. First, with a large number of hosts (more than about 20 or 30), you may want to organize administrative information into multiple NetInfo domain levels to keep things more manageable. Second, large networks are frequently made up of several smaller physical networks distributed among different locations. This chapter describes how to set up a multilevel NetInfo domain hierarchy and how to configure a NetInfo network distributed among different locations.

Important: A small mistake made during the initial configuration of a large network can result in many hours of corrective work. Before beginning your work, read this chapter thoroughly, along with Chapter 1, ^aUsing This Manual to Plan Your Network;^o Chapter 2, ^aSetting Up a NetInfo Network;^o and Chapter 3, ^aNetInfo Networking.^o

Background

As you've already seen, each NeXT computer serves one or more NetInfo domains from a database identified by a name called a *tag*. The domains are organized into a hierarchy, and searches for information begin at the local domain and then pass up the hierarchy. Setting up a multilevel NetInfo domain hierarchy (one with more than two levels) involves making direct modifications to components of the NetInfo databases. Before you begin, you should know how information searches are performed, how domains are identified, and how the NetInfo domain hierarchy is configured.

Naming

There are two different ways to identify a specific domain. The first, called *tagged domain notation*, specifies the host serving the domain and the tag of the database. This is usually written as *host/tag*, where *host* is either the host name or Internet address of the server, and *tag* is the tag of the domain being served.

The second convention is called *domain name notation*. Names in this format are similar to file pathnames. For example, ^/ is the root domain, and **/boston/earth** identifies the local domain on the host **earth**, which is a client of the midlevel domain **boston**. Two relative domain names are used, also similar to file names: ^.. is the current domain and ^.. is the parent domain. When you use an Open or Save domain panel, domains are identified using domain name notation.

Information Access

The following diagram shows the NetInfo domains available to seven computers that are hosts on a large network with a three-level NetInfo domain hierarchy. Each rectangle represents a domain. The word in each rectangle is the name of the domain, and the word to the left of each rectangle is the tag of the database for that domain. This diagram gives no indication of which computer is serving which domain (except the local domains).

F0.eps ,

Notice that each domain has exactly one parent (except the root domain) and zero or more children. Any given computer has access to the information stored in its local domain, the root domain, and all intervening domains between the local domain and the root domain. For example, the host **pluto** has access to the information in the domains **pluto**, **teachers**, and ^/ (root), while the host **mars** has access to the information in its local domain (**mars**) and the root domain (^/). Notice that the hierarchy doesn't have to be balanced—**pluto** has access to three domain levels, while **mars** only has access to two.

When a process requests information from NetInfo, the search begins in the local domain, then continues up the domain hierarchy until the information is found or the root domain is reached. For example, let's say that the user

pat attempts to log into the computer **pluto**. The **loginwindow** process makes a request, searching for the account information for the user **pat**. The domain **pluto** (the local domain) is searched first. If the account isn't found there, the request is passed to the domain **teachers**. If the account isn't found there, the request is passed to the domain ^{a/o} (root). If the account for **pat** is found in any domain, the search ends.

Binding

When a networked NeXT computer boots, it links the domains it serves into the hierarchy of NetInfo domains. This process is called *binding*, and is controlled by the **serves** property of host entries in the NetInfo directory **/machines**.

The serves Property

Follow these steps to examine the various **serves** properties (if you don't yet have a network configured, just examine the local domain):

1. Start up NetInfoManager.
2. In the local domain window, click **/machines**, then **broadcasthost**. Double-click **broadcasthost** to open the Directory window.
3. Click **serves**.

F5.tiff ,

Notice the value **../network**. This means that the host **broadcasthost** serves the parent domain (..) from a database tagged **network**.

4. Click **ip_address**.

F6.tiff ,

The value **255.255.255.255** is the broadcast address, meaning that any messages sent to this address will be sent to all computers on the local network. See Appendix C, ^aInternet Addressing,^o for more information.

5. Close the Directory window and double-click **localhost** in the local domain window. Click **serves** in the Directory window.

F7.tiff ,

Notice the value **./local**. This indicates that the host **localhost** serves the current domain (.) from a database tagged **local**.

6. Click **ip_address**.

F8.tiff ,

The value **127.0.0.1** is the loopback address, meaning that any messages sent to this address will be handled by the local computer without being sent to the network.

7. Choose Open from the Domain menu and open the root domain. Click **machines**.
8. Click the host name of the root domain server under the **/machines** directory, then double-click it to open the Directory window.
9. Click **serves**.

F9.tiff ,

The value **./network** indicates that this host serves the current domain (.) from a database tagged **network**. The value *domain*/**local** means that this host also serves the domain *domain* from a database tagged **local**.

Important: The name of a domain used in domain name notation is not determined by anything in the domain itself, but rather by the **serves** property of the host entry in the *parent* domain. By default, local domains are given the same name as the host serving the domain.

Example Properties

The following diagram shows three computers in a two-level domain hierarchy. The diagram includes the domains each computer serves (identified with both the database tag and the domain name).

F1.eps ,

In this example, **earth** is the master server for the root domain, and **venus** is a clone server for the root domain. A clone server provides read-only access to a domain. In this case, **venus** stores and serves an exact copy of the root domain from a database tagged **network**. The master server of a domain is identified by the property **master** in the root (^{a/o}) NetInfo directory of that domain. Here, the **master** property has the value **earth/network** (notice the tagged domain notation), indicating that **earth** is the master server for the root domain and serves the domain from the database tagged **network**. The host **venus** can tell from this property that it is a clone server and **earth** is the master.

The various **serves** properties identify the parent and children of the domains. Each local domain has identical host entries for **localhost** and **broadcasthost**. The **serves** property for **localhost** indicates that it serves the current (local) domain (.) from a database tagged **local**. The **serves** property for **broadcasthost** indicates that it serves the parent domain (..) from a database tagged **network**.

The root domain contains host entries for each of the three computers. From these entries, you can see that **mars** serves the domain **mars** from a database tagged **local**. The **serves** property for **venus** indicates that it serves the domain **venus** from a database tagged **local** as well as the current domain (.) from a database tagged **network**. The **serves** property for **earth** shows that it also serves the current domain from a database tagged **network**, as well as the domain **earth** from the database tagged **local**. The **master** property indicates **earth** is the master server of the current (root) domain, which means **venus** must be a clone server.

The Binding Process

Here's a detailed description of how the **serves** properties are used to bind a domain to its parent:

1. As the computer boots, a line in **/etc/rc** starts the NetInfo binding daemon, **nibindd**.

2. The **nibindd** daemon looks in **/etc/netinfo** for directories with names in the form *tag.nidb*. For each directory **nibindd** finds, it starts a **netinfod** daemon to serve data from the database. Each **netinfod** daemon is run with an argument matching the tag for the NetInfo database it's serving.

For example, **mars** has a directory named **/etc/netinfo/local.nidb**, so a **netinfod** daemon is started with the tag **local**.

3. Each **netinfod** daemon examines its **/machines** NetInfo directory for entries that include a **serves** property with a value in the form *../tag*. This value indicates that the host whose directory this is serves the parent NetInfo domain (*..*) from a database tagged *tag*.

On the host **mars**, the entry **/machines/broadcasthost** has a **serves** property with the value **../network**. It's the only entry with a value of the **serves** property in the form *../tag*.

4. A special message called a *bind request* is sent to the Internet address associated with each **/machines** entry with an appropriate **serves** value (*../tag*). The bind request includes the sender's Internet address, the tag of the domain looking for its parent, and the tag of the parent domain.

The value of the **ip_address** property for **broadcasthost** is **255.255.255.255**, so the bind request is broadcast to all the hosts on the local network.

5. Each host that receives the bind request passes it to **nibindd**, which checks to see if there is a **netinfod** running for the requested parent domain tag. If there is, the request is forwarded to the appropriate **netinfod** process.

The **nibindd** processes on **earth** and **venus** each find that there is, indeed, a **netinfod** process running for the tag **network**. The bind request is passed on to the **netinfod** process on both servers.

6. The receiving **netinfod** process checks the **/machines** NetInfo directory for entries that have an **ip_address** property with a value that matches the sender's Internet address *and* a **serves** property with a value of the form *domain/tag*, where *tag* matches the tag of the domain looking for its parent.

The **netinfod** processes on both **earth** and **venus** find the entry **/machines/mars** with an **ip_address** property that has a value matching the Internet address of **mars**. The host entry in each case also includes a **serves** property with the value **mars/local**.

7. If the receiving **netinfod** process finds a host entry with appropriate values, a response is sent to the sender's Internet address indicating that the receiving host can serve the parent domain.

The hosts **earth** and **venus** each send a response to **mars** indicating that they can serve the parent domain.

8. The domain looking for its parent binds to the first server to respond.

Assuming that **earth** was busy responding to other NetInfo requests, **venus** might have responded first. In that case, **mars** binds to **venus** as its parent domain server.

9. The child domain searches the parent domain for the Internet addresses of any other hosts that serve the parent domain. These are identified by a **serves** property with the value *./parent_tag*, where *parent_tag* is the tag of the parent domain. The addresses are stored for possible future use.

When the search is made through the **/machines** directory in the root domain, the entries **venus** and **earth** are identified as having **serves** properties with the value **./network**.

10. If a request for information is later sent to the parent domain, and the server doesn't respond within a specific period of time, the child domain once again sends a bind request. This time, the request is sent to the Internet addresses of the hosts that are listed as serving the parent domain.

If **venus** doesn't respond quickly enough to a request, **mars** sends a new bind request to both **venus** and **earth**.

Planning

As with any endeavor, it's important to plan your network before beginning configuration. Here are some things to consider when planning for a large network:

- **Equipment**—If your network is made up of multiple physical networks (each called a *subnet*), you'll need additional equipment. Specifically, you'll need routers (or gateways) to route messages from one network to the next, and some kind of connection between routers. These connections might be, among other things, cables strung through the walls, buried fiber-optic cables between buildings, or dedicated phone lines leased from your phone company. See Chapter 1 for more information.
- **Addresses**—Make sure you have an appropriate network address. It's highly recommended that you register

your address and Internet Domain name with the Network Information Center. If you'll be using subnets, decide on a subnet mask and assign a network address to each subnet. For more information on obtaining a network address and subnetting, see Appendix C, ^aInternet Addressing.^o

- NetInfo Domains—Decide if you need a three-level domain hierarchy, or if you need one with even more levels (more than three levels is rarely necessary). Choose a name for each midlevel domain, then decide which host will serve each domain, which hosts will be clone servers, and which hosts will be clients. Decide in which domain user accounts, aliases, NFS mounts, and other administrative information should be stored. If you'll be using subnets, you'll probably want a domain for each subnet.

Considerations for Subnets

If your network is configured using subnets, several points to consider are:

- Broadcast messages aren't passed on by routers. Since NeXT computers get their host name and Internet address at boot time by sending a broadcast message, make sure that each subnet has a configuration server. See ^aCreating New Midlevel Domains^o later in this chapter.
- Information requests to the root domain can be frequent, and traffic over the router is slower than local traffic. It's a good idea to have a clone server for the root domain in each subnet. See ^aCreating Clone Servers^o later in this chapter.
- If you have configured network time service (and you should), consider having a clone time server in each subnet. You can have the midlevel domain server be the clone time server as well. For details, see Chapter 3.

Previewing the Result

When you set up a multilevel domain hierarchy, the most critical aspect is the location and values of host entries

and **serves** properties. Five critical rules that you must follow when you create your new hierarchy are:

- Each domain *must* have a host entry for each of its children (computers that serve a domain that binds to the current domain as its parent). Each host entry must have a **serves** property with the value *domain/tag*, where *domain* is the name of the child domain and *tag* is the tag of the child domain database.
- Each domain *must* have a host entry for the server of its parent domain. This host entry must have a **serves** property with the value *../tag*, where *tag* is the tag of the parent domain database. If there are any clone servers for the parent domain, there must be a host entry for each clone server with the same **serves** value (unless **broadcasthost** is used to identify the parent domain servers).
- Each domain *must* have a host entry for each clone server of the domain. These host entries must have a **serves** property with the value *./tag*, where *tag* is the tag of the current domain database.
- If a domain has host entries for computers that aren't serving either the parent domain or a child domain, the host entries must *not* include the **serves** property. Inappropriate **serves** properties can cause a domain to bind to the wrong parent, or not bind at all.
- In order for your hosts to communicate with each other, they must each have access to host entries for the other computers. The root domain should have a host entry for each computer on the network. Except for midlevel and root domain servers, these host entries must not have a **serves** property.

Another way to state these rules is: in the host entries of any given domain, there should only be **serves** properties for the parent domain (*../tag*), the current domain (*./tag*), and the children of the current domain (*domain/tag*). The root domain doesn't have a parent, and local domains don't have children.

The following diagram shows the same three computers as in the previous diagram. However, they are now shown after having been configured into a three-level domain hierarchy as preparation for creating the large network of seven computers shown in the first diagram. As you examine the diagram, notice how the modifications to the domains follow the host entry rules. It's a good idea to write down how the host entries will be set up in your new hierarchy, or at least have a mental picture of the end result.

F2.eps ,

In this diagram, **earth** serves the new root domain from a database tagged **super**, and the former root domain is now named **students**. The host **earth** is still the master server of the midlevel domain, and **venus** is still a clone

server of that domain.

The local domain on each computer hasn't been changed. The host entries have the same **serves** properties as before.

The midlevel domain **students** (the former root domain) has had a few modifications made. The host entry for **mars** has been deleted, since **mars** now binds to the new root domain as its parent. The host entry for **venus** is exactly the same. **venus** serves its local domain (**venus**) from the database tagged **local** and the current domain (.) from the database tagged **network**. The **serves** property in the host entry for **earth** still has the two values indicating that **earth** serves **earth/local** and **./network**. A third value has been added indicating that **earth** serves the parent domain (..) from a database tagged **super**. The master of this domain is still **earth**.

The new root domain includes host entries for all three computers. The **serves** property for **mars** shows that it only serves its local domain. The entry for **venus** includes a **serves** property indicating that it serves the domain **students** from a database tagged **network** (this is the former root domain). The entry for **earth** shows that it also serves **students** from the database **network**, as well as serving the current domain (.) from a database tagged **super**. The **master** property shows that **earth** is the master server for this domain as well. No clone has yet been configured for the new root domain.

Overview of Procedures

How you set up your network depends on what you're starting with. You might be starting from scratch and setting up the entire network at once. You might already have several small networks and now want to connect them together into a large network. Or, you might have a single network that you want to organize into several NetInfo domains. This section provides a general description of the procedures to follow in each situation. References are made to complete instructions, which are all in the section ^aBuilding the Hierarchy^o later in this chapter.

Warning: Do *not* reboot any computer except when the procedures say to do so. Rebooting at the wrong time will make domains bind to the wrong parent or not bind at all.

Starting from Scratch

If you don't have any existing networks, the easiest procedure is set up several small networks and then combine them into one domain hierarchy. Even if your network won't be made up of physically separate networks, configure each midlevel domain and its clients as a separate network, then connect them and create the top-level domain.

These are the procedures to use if you're starting from scratch:

1. Use SimpleNetworkStarter to configure several small networks, following the instructions in Chapter 2, ["Setting Up a NetInfo Network."](#) The master server of each small network will eventually be a midlevel domain server. Configure each small network to include the computers that you want to be children of that midlevel domain. If each smaller network will be a subnet, configure routing and set a netmask as you build each network. Follow the instructions in ["Configuring Routing and Setting a Netmask."](#)
2. Create a new top-level domain and rename the former root domain of the current small network, following the procedures in ["Creating a New Root Domain."](#)
3. Incorporate the root domain of each of the other smaller networks into the new hierarchy as midlevel domains. Follow the instructions in ["Incorporating Midlevel Domains."](#)
4. Copy host entries from the midlevel domains to the new root domain and delete various **serves** properties. See the section ["Transferring Host Entries to the Root Domain."](#)
5. Reboot the computers.
6. Create appropriate clone servers, following the procedures in ["Creating Clone Servers."](#)

Multiple Existing Networks

If you already have existing networks, simply connect them together and create a new root domain. These are the procedures to follow if you're starting with multiple networks:

1. If each existing network will be a subnet, use HostManager to configure routing and set a netmask. Follow the instructions in ["Configuring Routing and Setting a Netmask."](#)
2. Create a new top-level domain and rename the former root domain of the current small network, following the

procedures in [Creating a New Root Domain](#).

3. Incorporate the root domain of each of the other existing networks into the new hierarchy as midlevel domains. Follow the instructions in [Incorporating Midlevel Domains](#).
4. Copy host entries from the midlevel domains to the new root domain and delete various **serves** properties. See the section [Transferring Host Entries to the Root Domain](#).
5. Reboot the computers.
6. Create appropriate clone servers, following the procedures in [Creating Clone Servers](#).

Single Existing Network

If you already have a single network that you want to reconfigure into a multilevel domain hierarchy, you can create a new root domain and then create new midlevel domains. These are the procedures to follow in this situation:

1. Create a new top-level domain and rename the former root domain, following the procedures in [Creating a New Root Domain](#).
2. Create new midlevel domains, rename them, then make each midlevel domain server a configuration server. Copy host entries from the former root domain to each of the new midlevel domains. Follow the procedures in [Creating New Midlevel Domains](#).
3. Copy host entries from the former root domain to the new root domain and delete various **serves** properties. See the section [Transferring Host Entries to the Root Domain](#).
4. Remove duplicate host entries from the former root domain. Follow the procedures in [Deleting Host Entries](#).
5. Reboot the computers.
6. Create appropriate clone servers, following the procedures in [Creating Clone Servers](#).

Building the Hierarchy

This section includes detailed instructions for building a network with a multilevel NetInfo domain hierarchy. You don't need to perform all the procedures described here—only some will apply, depending on your starting point. Be sure to read the previous section carefully and only follow those procedures appropriate to your situation.

Configuring Routing and Setting a Netmask

If you're using subnets, you need to configure routing, assign a network address to each subnet, and configure each subnet to use a subnet mask. See Appendix C for information on routing, choosing subnet addresses, and choosing netmasks. If you're not using subnets, skip this section.

Using SimpleNetworkStarter

The procedures in this section are appropriate if you're starting from scratch. To assign a netmask and set up routing as you build the network, follow these steps:

1. Log into the host that will be the master server for the subnet.
2. Start up SimpleNetworkStarter.
3. Click the button labeled "Be a server...". Enter the host name and Internet address into the fields in section 2 of the window.
4. Click the Other Options button. The Other Network Information panel appears.

F10.tiff ,

5. In the Router section, click the button labeled Dynamic.

Note: If the subnet uses a single router, you can eliminate unnecessary network traffic by clicking the button

next to the text field in the Router section and entering the Internet address for the router.

6. In the Netmask section, click the button next to the text field and enter your netmask. The netmask must be entered in the same format as an Internet address.

F11.tiff ,

In this example, the value **255.255.255.0** assigns the last 8 bits of the Internet address to the host portion.

7. Click OK.
8. Complete the configuration of the server following the instructions in Chapter 2.
9. Start up HostManager and choose Automatic Host Configuration from the Network menu.
10. Modify the address fields to make sure hosts are assigned addresses that conform to your subnet address.
11. Add the remaining hosts to this subnet following the instructions in Chapter 2.
12. Set up clone servers following the instructions in Chapter 2. As you configure each clone with SimpleNetworkStarter, use the Other Options button to set Routing to Dynamic and Netmask to Automatic.
13. Log into each remaining host in turn. Start up SimpleNetworkStarter and click "Be a client...". Click Other Options and, in the panel that appears, set the Netmask to Automatic and the Router to Dynamic. Again, if the subnet uses a single router, enter the Internet address of the router in the text field instead of setting Router to Dynamic. Click Configure this Client.

Note: If you prefer, you can use the Local Configuration window in HostManager to set routing and the netmask for each client.

14. Repeat these steps for each subnet.

Using HostManager

You can't use SimpleNetworkStarter to assign a netmask and configure routing on an existing server. Instead, you use HostManager. These procedures are appropriate if you're starting with multiple existing networks.

1. Log into the master NetInfo server for the existing subnet.
2. Start up HostManager and choose Local from the main menu.
3. In the Router section, click the button labeled Dynamic.

Note: If the subnet uses a single router, you can eliminate unnecessary network traffic by clicking the button next to the text field in the Router section and entering the Internet address for the router.

4. In the Netmask section, click the button next to the text field and enter your netmask. The netmask must be entered in the same format as an Internet address.

F12.tiff ,

In this example, the value **255.255.255.0** assigns the last 8 bits of the Internet address to the host portion.

5. Click Set.
6. Log into each client in turn. Start up SimpleNetworkStarter and click "Be a client...". Click Other Options and, in the panel that appears, set the Netmask to Automatic and the Router to Dynamic. Again, if the subnet uses a single router, enter the Internet address of the router in the text field instead. Click Configure this Client.

Note: If you prefer, you can use the Local Configuration window in HostManager to set routing and the netmask for each client

7. Make sure each host's Internet address conforms to the network address assigned to the subnet and is unique from addresses used in any other subnet. Use HostManager to change any duplicates, following the instructions in Chapter 3.

Creating a New Root Domain

This section describes how to create a new top-level domain. These procedures apply in all situations.

Creating the Top-level Domain

Your first step is to create the new domain.

1. Log into the new root domain server as **root** (this must be one of the existing root domain servers). You *must* log in as **root** to use NetInfoManager to create a domain. In our example, you would log into **earth**.
2. Start up NetInfoManager.
3. Choose New from the Domain menu. The New Domain Creation panel appears.

F13.tiff ,

4. Click the button labeled "Create a new domain which will be the new root domain."

F14.tiff ,

5. Enter the tag for the new domain in the Domain Tag text field. The tag can be anything you want, except **local** or **network**. In the example, the tag is **super**.

F15.tiff ,

6. Click Create. After a few seconds, a new domain window appears.

F16.tiff ,

Notice that the new domain is created with standard directories.

Renaming the Midlevel Domain

Now you need to rename the former root domain (now a midlevel domain) to match the name you've chosen.

1. In the new root domain window, click **machines**.

F17.tiff ,

2. Click the only name in the **/machines** directory (**earth** in our example). This is the host entry for the former root domain server of this subnet. Double-click the name to open a Directory window.
3. Click **serves**.

F18.tiff ,

Notice that there are two values associated with the **serves** property. The first is *./tag*, where *tag* is the tag you assigned to the new domain, indicating that this host serves the current (new root) domain from a database tagged *tag*. The second is *domain/network*, where *domain* is the host name of the former network server, indicating that this host serves the domain *domain* (the former root domain) from a database tagged **network**.

In the example, these value are **./super** and **earth/network**.

4. Click the value *domain/network* and replace *domain* with the new name you've chosen for this midlevel domain. (In the example, you would change **earth/network** to **students/network**.) Press Return, then choose Save from the Directory menu.

F19.tiff ,

Note: If you don't change the domain name, your domain hierarchy will look like */host/host* (**/earth/earth**, for example).

5. Open the former root domain by choosing Open from the Domain menu and double-clicking ^{a/o} in the panel that appears. The new root domain won't be incorporated into the hierarchy until you reboot the server, so ^{a/o} in the Open Domain panel is the old root domain.
6. Click **machines**, then the host name of the computer you're logged into (**earth** in the example).

7. Double-click the host name to open the Directory window. Click **serves**.

F20.tiff ,

Notice the three values *domain/local*, *./network*, and *./itag*. The first two values were there before, but the last is how the former root domain will bind to the new root domain as its parent. In the example, these are **earth/local**, **./network**, and **./super**.

8. Reboot this computer.

Important: The new root domain doesn't contain a **root** user account. Without this account, you can only make modifications to the new root domain while logged in as **root** on the server. Use UserManager to add a **root** account to the new root domain.

Incorporating Midlevel Domains

This section describes how to incorporate former root domains into the new domain hierarchy as midlevel domains. These procedures are appropriate if you either started from scratch or began with multiple networks. If you began with a single network, skip this section.

You now have a three-level hierarchy with a single midlevel domain (one of the former root domains). The next step is to incorporate the other root domain servers as new midlevel domain servers. First, add a host entry to the root domain for each midlevel domain server, then add a host entry to the midlevel domains for the new root domain server.

Identifying the Midlevel Domains in the Root Domain

Add a host entry in the new root domain for each of the midlevel servers, including any clone servers of the former root domains. For example, if the root domain server of one of your subnets is **pluto**, and **neptune** is a clone server for that domain, add a host entry for each.

1. Log into the new root domain server and start up NetInfoManager.

2. Chose Open from the Domain menu and open the new root domain (since you rebooted the server, it's now incorporated into the hierarchy as ^{a/o}).
3. Select the **/machines** directory in the domain window.
4. Click the only subdirectory (**earth** in the example) and make a copy by choosing Duplicate from the Edit menu.

F23.tiff ,

5. Double-click the new directory to open a Directory window.

F22.tiff ,

6. Change the value of the **name** property to the host name of one of the new midlevel domain servers (former subnet root domain server. In the example, this might be **pluto**). If the server has any host aliases, add them as additional values of the **name** property.
7. Change the value of the **ip_address** property to be the correct Internet address for the host.
8. Click **serves**. Change one value to be *domain/network*, where *domain* is the name you've chosen for this midlevel domain. For **pluto**, this would be **teachers/network**.
9. Delete any other values for the **serves** property.

F21.tiff ,

10. Choose Save from the Directory menu.
11. Repeat these steps to add a new host entry to the root domain for each of the other midlevel servers, including any existing clone servers.

Identifying the Parent to the Midlevel Domains

Next, add a host entry for the new root domain server in each of the midlevel domains:

1. Log into the midlevel domain server and start up NetInfoManager. (The midlevel domain server is the former subnet root domain server. In the example, this might be **pluto**.)
2. Choose Open from the Domain menu and select ^{a/o} in the panel that appears. The new root domain won't be identified from the other subnets until you've made all your modifications and rebooted the midlevel servers, so ^{a/o} in the Open Domain panel is still the old root domain of the subnet.

Note: Assuming you've connected all your computers together and configured any necessary routing, you can make these modifications from the server of the new root domain. Choose Open by Tag from the Domain menu, then enter the Internet address of the midlevel domain server into the Host field, and enter **network** into the Tag field. Click Open.
3. Select **/machines** in the domain window, then click one of the host entries. Choose Duplicate from the Edit menu.
4. Double-click the new directory to open a Directory window.
5. Modify the property values to hold the host name, Internet address, and Ethernet address of the server of the new root domain (in our example, these are the values for **earth**).
6. Modify the value of the **serves** property to be **../tag**, where *tag* is the tag you assigned to the new root domain. Delete any other values for the **serves** property.
7. Choose Save from the Directory menu.
8. If you haven't yet done so, physically connect this subnet to the subnet containing the new root domain server.
9. Reboot the new midlevel domain server.
10. Repeat these steps on each of the new midlevel domain servers (former root domain servers).

Creating New Midlevel Domains

If you began your work with a single network, follow the procedures in this section. If you started from scratch or began with multiple existing networks, skip this section.

You now have a three-level hierarchy with a single mid-level domain (the former root domain). The next step is to create the other midlevel domains.

Creating a Midlevel Domain

The first step is to create the new domain.

1. Log into the new midlevel domain server as **root**. You *must* log in as **root** to use NetInfoManager to create a domain.
2. Start up NetInfoManager. Choose New from the Domain menu. The New Domain Creation panel appears.

F24.tiff ,

3. Click the button labeled ^aCreate a domain as a child of an existing network domain.^o

F25.tiff ,

4. Click Select Parent. Select the root domain in the Select Netinfo Domain panel that appears, then click OK.

F26.tiff ,

5. Enter **network** into the Domain Tag field.
6. Click Create. After a few seconds, a new domain window appears.

F27.tiff ,

Renaming the Midlevel Domain

Now you need to rename the new midlevel domain so that it matches the name you've chosen.

1. Open the root domain and click **machines** in the domain window.

F28.tiff ,

2. Click the host name of this midlevel domain server in the **/machines** directory (**pluto** in our example). Double-click the name to open a Directory window.
3. Click **serves**.

F29.tiff ,

Notice the value *domain/network*, where *domain* is the host name of this midlevel domain server, indicating that this host serves the domain *domain* from a database tagged **network**.

In the example, this would be **pluto/network**.

4. Click the value *domain/network* and replace *domain* with the new name you've chosen for this midlevel domain. (In the example, you would change **pluto/network** to **teachers/network**.) Press Return, then choose Save from the Directory menu.

Note: If you don't change the domain name, your domain hierarchy will look like */host/host* (**/pluto/pluto**, for example).

5. Click **machines** in the new midlevel domain window, then the host name of the root domain server (**earth** in the example).
6. Double-click the host name to open the Directory window. Click **serves**.

F30.tiff ,

Notice the value *../tag*. This is how the new midlevel domain will bind to the root domain as its parent. In the

example, this is **../super**.

7. Close the Directory window, then double-click the name of the midlevel domain server in the new midlevel domain window to open another Directory window. Click **serves**.

F31.tiff ,

8. Click the value, then choose Append Value from the Directory menu.
9. Click **new_value** and enter *host/local* in the text field, where *host* is the host name of this midlevel domain server. This indicates that this host also serves the domain *host* from the database tagged **local**. In the example, this would be **pluto/local**.
10. Press Return, then choose Save from the Directory menu.

F32.tiff ,

Transferring Host Entries to Midlevel Domains

Now transfer host entries from the former root domain to the new midlevel domain. These host entries are for the clients (children) of the new midlevel domain server.

1. Click **/machines** in the new midlevel domain window (in the example, **teachers**).
2. Choose Open from the Domain menu and open the former root domain (now a midlevel domain—in the example, **students**).
3. Click **/machines** in the former root domain window that appears.
4. Drag the former root domain window (**students**) aside so you can see both it and the new midlevel domain window (**teachers**).
5. In the former root domain window, click the name of the host entry for one of the computers that will be a client of the new midlevel domain.
6. Drag the folder icon from the well in the former root domain window to the well in the new midlevel domain

window. This makes a copy of the host entry in the midlevel domain.

7. Repeat, copying the host entries for each of the clients from the root domain to the midlevel domain.

Making the Midlevel Domain Server a Configuration Server

Since you're creating a new domain on a computer that wasn't configured as a server with SimpleNetworkStarter, it won't automatically be set up as a configuration server. To set up a computer as a configuration server, follow these steps:

1. Start up HostManager on the midlevel domain server.
2. Choose Local from the main menu. The Local Configuration window appears.

F33.tiff ,

3. Click the button next to the text field under Hostname, then click the button next to the text field under Internet Address. The host name and Internet address are automatically entered into the text fields.

F34.tiff ,

4. Click Set. In the panel that appears, click Skip Reboot.
5. Open the file **/etc/hostconfig**.
6. Edit the value of the variable NETMASTER to be equal to -YES-.
7. Reboot the computer.

This computer will now get its host name and Internet address from **/etc/hostconfig** instead of broadcasting to a configuration server. It will also now respond to configuration requests from other computers.

Repeat these procedures for each midlevel domain server, beginning with ^aCreating a New Midlevel Domain.^o

Important: The new midlevel domains don't contain a **root** user account. Without this account, you can only

make modifications to a new midlevel domain while logged in as **root** on the server. Use UserManager to add a **root** account to the new midlevel domains.

Transferring Host Entries to the Root Domain

This section describes procedures that are appropriate in all situations.

Transferring the Host Entries

In order for the computers on your network to communicate with each other, they need access to a host entry for each of the other computers. By creating host entries in the root domain, you make them available to the entire network.

1. Start up NetInfoManager and open the new root domain.
2. Click **/machines**.

F35.tiff ,

3. Open one of the former root domains (now a midlevel domain). There's only one if you began with a single network (**earth**).
4. Click **/machines** in the midlevel domain window.
5. Click the name of a host that's *not* a midlevel domain server or the root domain server.
6. Drag the midlevel domain window away, so you can see both this window and the root domain window.
7. Drag the folder icon from the well in the midlevel domain window to the well in the root domain window. This makes a copy of the host entry in the root domain.

F36.tiff ,

8. Make a copy in the root domain of all other host entries in the midlevel (former root) domain that aren't midlevel or root domain servers.
9. If you started from scratch or began with multiple networks, repeat these procedures from step 3 for each of the other former root domains.

Deleting serves Properties

The root domain now has a host entry for every computer on the network. Only host entries for midlevel or root domain servers should have a **serves** property. Now delete the **serves** property for all other host entries in the root domain.

1. Open the root domain, if it's not already open.
2. Click **/machines**, then the name of a host that is neither a midlevel nor root domain server.
3. Double-click the name to open a Directory window. Click **serves**.

F37.tiff ,

4. Choose Delete from the Edit menu. Having a **serves** property here may cause the local domain to bind to the root domain instead of a midlevel domain.

F38.tiff ,

5. Delete all other properties *except* **name** and **ip_address**. The other properties aren't necessary in the root domain.
6. Choose Save from the Directory menu.
7. Delete all properties except **name** and **ip_address** from all other host entries that are neither the root domain server nor a midlevel domain server.
8. If you started from scratch or began with multiple networks, reboot the computers. Begin with the root domain server, then the midlevel domain servers, then the clients.

Deleting Host Entries

If you started with a single network, you need to eliminate duplicate host entries in the former root domain. Skip this section if you began with any other situation. The former root domain still contains host entries for computers that are now clients of other midlevel domains. It's important to delete these entries to prevent clients from binding to the wrong parent.

1. Open the former root domain (now a midlevel domain).
2. Click **/machines**, then the name of a host that is now a client of one of the other midlevel domains.
3. Choose Delete from the Edit menu. Click Delete Anyway in the panel that appears.

F39.tiff ,

4. Delete all other host entries for computers that are now clients of another midlevel domain server.
5. Reboot all the computers, beginning with the servers.

Creating Clone Servers

Now that you have a three-level domain hierarchy, it's a good idea to create at least one clone for the root domain, and at least one for each midlevel domain. If you're using subnets, you should have a clone of the root domain in each subnet.

1. Log into the computer that will be the clone server as **root**.
2. Start up NetInfoManager, then choose New from the Domain menu.
3. Click ^aClone an existing domain.^o

4. Click Select Domain. The Select NetInfo Domain panel appears.

F40.tiff ,

5. Select the appropriate domain in the browser, then click OK.

F41.tiff ,

6. Click Create.

Tip: If you're cloning the root domain, it's a good idea to make sure that each midlevel server recognizes the clone server. You do this by making sure the midlevel domain has a host entry in **/machines** for the clone server. This entry should include a **serves** property in the form *../tag*, where *tag* is the tag of the root domain. Without this host entry, the midlevel domains will only bind to the master root domain server when they boot (although they might bind to a clone server later). If the master domain server isn't available when the midlevel domain servers boot, they won't bind at all.

Transferring NetInfo Data

If you're working with one or more existing networks, you might have user accounts, aliases, user groups, NFS directory mounts, or other information in one or more of the existing domains. With your new three-level domain hierarchy, you should move this data to the appropriate domain. Remember that a given computer has access to data in the local domain and successive parent domains up to the root domain.

For example, a user account stored in the root domain allows that user to log into any computer on the network. That same account stored in a midlevel domain restricts the user to the computers served by that midlevel domain. Make sure that a user's home directory is available (with NFS) on every computer that has access to the account information. Be very careful about conflicts with user IDs, group IDs, names, mounting information, and so on. A thorough planning session can prevent a lot of confusion.

Troubleshooting

This section contains troubleshooting hints for some of the problems that might arise after configuring a large network.

Binding Problems

When you boot a computer, you might see the following message:

```
Still searching for parent network administration (NetInfo) server.  
Please wait, or press `c' to continue without network user accounts.  
See your system administrator if you need help.
```

This message is displayed when your computer is looking for the parent of its local domain, and the NetInfo server for the parent domain doesn't respond. If you type **c** at this point, your local domain isn't bound to its parent domain, and all configuration information residing in that domain (including users, aliases, and machines) is not available.

This message can occur for various reasons:

- All servers for the parent NetInfo domain are unavailable.
- Your computer's Internet address is not recognized by a server for its parent NetInfo domain.
- The local domain is referencing a computer as a server for its parent domain which is not, in fact, such a server.
- The local domain is referencing a nonexistent computer as the one to serve its parent NetInfo domain.
- Although servers for the parent NetInfo domain are running, no servers responded to the request.

A server for a domain might not respond to a request for one of two reasons:

- The server or the network is overloaded.

- The server is ignoring all NetInfo requests.

When a NetInfo server is started, it checks the consistency of its database; during this operation, it doesn't respond to NetInfo requests. If the system crashed, and if you have a large NetInfo database or large NetInfo network, it can take a long time (several minutes) to complete these consistency checks. When the checks are completed, the server for the parent domain responds to your computer's request, and your computer will bind correctly to its parent NetInfo server.

If nothing is really wrong, the boot should eventually continue without intervention (within no more than 20 minutes). If you press **c**, you continue the boot process as if you were not connected to a network, ignoring any domains above your local domain.

If the boot process doesn't continue, press **c**, then check the host entries and **serves** properties for all the relevant hosts in all the relevant domains. Make sure the parent domain has a host entry with the proper **serves** property for the child domain server. Make sure the child domain has a host entry with an appropriate **serves** property for the parent domain. Make sure that the host entries for computers that aren't parent domain servers don't have **serves** properties for the parent domain. Make sure other domains don't have a host entry with a **serves** property for the child domain server.

Routing Messages

If your domain is on a subnet and your router is unavailable, you might see a message similar to the following:

```
Cannot send many-cast packet to Internet_address, retrying
```

This is a result of the normal binding process, and when your router becomes available again, the binding process will continue.

Communication Problems

If your computers boot successfully, but some can't seem to communicate with others, you may have host entry problems. Use **ping** to see if one host can communicate with another. If you get a message such as ^aunknown

host,^o check the host entries in the root domain. There should be a host entry for every computer on the network. Make sure the host entry includes any host name aliases for that host.

If all the host entries seem to be in order, but some hosts still can't communicate, you might have a netmask problem.

Netmask Problems

The netmask determines which bits of an Internet address are used to identify the network and which bits identify the host. If one or more computers are using different subnet masks, the hosts won't be able to communicate with each other. Use HostManager to check that each server has the correct netmask value set, and that each client has Netmask set to Automatic. Correct any inconsistencies. See Appendix C for more details about subnet masks.