17

Administrative Services: DNS, FTP, and Logging

CERTIFICATION OBJECTIVES

This chapter examines four administrative services: the Domain Name System (DNS), system activity reports, system logging services, and the Network Time Protocol (NTP) service. That's less complex than it sounds, as for DNS, the RHCE objectives require only the configuration of a caching-only nameserver that may forward queries. So this chapter does not cover the configuration of a master or secondary DNS server.

Linux system utilization reports are associated with the **sar** command, which is relatively easy to configure as a cron job. While basic logging on a local system was described in Chapter 9, this chapter covers the configuration of a central logging server. Finally, while NTP clients were described in Chapter 5, this chapter covers the configuration of an NTP server.

INSIDE THE EXAM

While this chapter addresses several RHCE objectives, the steps required to meet each of these objectives are fairly simple, relative to the tasks associated with the configuration of file sharing services such as Samba, NFS, and vsFTP.

Domain Name Service

Examine the RHCE objectives associated with DNS:

- Configure a caching-only name server
- Configure a caching-only name server to forward DNS queries

This is less than was required in previous versions of the RHCE objectives, which specified the configuration of a slave (or secondary) name server. Red Hat has addressed that change in the latest objectives, with the following statement:

"Note: Candidates are not expected to configure master or slave name servers."

As the only SELinux targeted setting relates to the overwriting of master zone files, it is not an issue for RHCE-related DNS configuration.

System Utilization Reports

While system utilization reporting is a new requirement for the RHCE, it's an important skill for all computer professionals. As an RHCE, you'll be expected to have the answers for a variety of computer-related questions. Since RHEL 6 includes the sysstat package for such reports, the related objective, as follows, makes sense:

- Produce and deliver reports on system utilization (processor, memory, disk, and network)

As this is not a network objective, there is no need to configure a firewall. There are no current SELinux-related system utilization options. However, as with other services, it's hard to see how you'd get credit for any configuration made for system utilization reports unless such services are booted automatically.

System Logging Server

Nominally, the standard RHEL 6 system logging service is a network server, like other network servers discussed in the second half of this book. But based on the way the RHCE objectives are written, system logging is an element of "System Configuration and Management." System logging information that's transmitted over a network should be configured on the client, to send logging information to a remote system, and the logging server should be configured to accept logging information from remote systems.

The system logging server normally uses TCP or UDP port 514. However, there are no current SELinux-related options. As system logging is the province of the root administrative user, there are no other real user limitations, unless you configure users in the /etc/sudoers file to administer the rsyslog service using the techniques discussed in Chapter 8. However, host limits are possible based on **iptables**-based firewalls.

The Network Time Service

Finally, the last service covered in the RHCE objectives is based on NTP. The requirement suggests that you need to know how to take advantage of the cooperative nature of NTP servers:

- Synchronize time using other NTP peers

CERTIFICATION OBJECTIVE 17.01

Basic Domain Service Organization

DNS is a service that translates human-readable domain names such as www.mheducation.com to IP addresses such as 12.163.148.101, and vice versa. DNS is a distributed database; each server has its own delegated zone of authority for one or more domains. The DNS service associated with RHEL is the Berkeley Internet Name Domain (BIND). As no individual DNS server is large enough to keep a

database for the entire Internet, each server is configured by default to refer requests to other DNS servers.

Basic Parameters

DNS on RHEL 6 is based on the **named** daemon, built on the BIND package developed through the Internet Software Consortium. RHEL 6 includes BIND version 9.7. You can use the **rndc** command to manage DNS operation; it's functionally similar to how you can use the **apachectl** command to manage the Apache web server.

DNS Package Options

To configure a system as a DNS server, you should be interested in the packages associated with the Network Infrastructure Server package group. As all packages in that group are options, you'll need to specify packages individually. With that in mind, you should know something about the RPM packages associated with DNS:

- **bind** Includes the basic name server software and extensive documentation.
- **bind-chroot** Adds directories that isolate BIND in a so-called "chroot jail," which limits access if DNS is compromised.
- **bind-devel** Includes development libraries for BIND.
- **bind-dyndb-ldap** Supports dynamic updates to LDAP.
- **bind-libs** Adds library files used by the bind and bind-utils RPMs.
- **bind-sdb** Supports alternative databases, such as LDAP.
- **bind-utils** Contains tools such as **dig** and **host** that provide information about a specific network host in some DNS database.

By now, you should be comfortable installing these packages with commands like **yum** from installation databases as discussed in Chapter 7.

on the job

RHEL 6 also supports the dnsmasq package, which can also be used to set up a forwarding DNS server.

Different Types of DNS Servers

While additional options are available, there are four basic types of DNS servers:

- A master DNS server, authoritative for one or more domains, includes host records for that domain.
- A slave DNS server, which relies on a master DNS server for data, can be used in place of that master DNS server.
- A caching-only DNS server stores recent requests like a proxy server. If configured with forwarding features, it refers to other DNS servers for requests not in its current cache.
- A forwarding-only DNS server refers all requests to other DNS servers.

Each of these servers can be configured with access limited to internal networks, or even just a local system. Alternatively, they can be configured as public DNS servers, accessible to the entire Internet. But such access comes with risks, as successful attacks against an authoritative corporate DNS server could easily keep their web sites hidden from customer's web browsers, a different form of denial of service.

CERTIFICATION OBJECTIVE 17.02

Minimal DNS Server Configurations

You can configure DNS servers by directly editing the associated configuration files. In this section, you'll briefly review the configuration files installed with the BIND software. You'll then learn how to configure a caching-only nameserver, as well as a nameserver that includes forwarding to specified DNS servers.

BIND Configuration Files

DNS configuration files can help you configure a Linux system as a database of hostnames and IP addresses. That database can be cached, listed in a local database, or the request can be forwarded to a different system. The configuration files that support the use of DNS as a server are described in Table 17-1. While the table includes references to standard /var/named database files, changes to such files are not required to configure a caching-only or forwarding DNS server.

TABLE 17-1 DNS Server Configuration Files

DNS Configuration File	Description
/etc/sysconfig/named	Specifies nonstandard configuration and data file directories.
/etc/named.conf	Notes the standard location for the main DNS configuration file. Can incorporate data from other files, normally in the /etc/named directory, with the include directive.
/etc/named.iscdlv.key	Specifies the standard DNS encryption key.
/etc/named.rfc1912.zones	Adds appropriate zones for localhost names and addresses.
/etc/rndc.key	Lists the authentication key required to support requests to the DNS server.
/var/named/named.empty	Includes a template zone file.
/var/named/named.localhost	Lists the zone file for the localhost computer.
/var/named/named.loopback	Lists the zone file for the loopback address.

If you've installed the bind-chroot package and have started the named service, a hard link to these files will also be available in the /var/named/chroot directory. That directory is configured through the ROOTDIR directive in the /etc/sysconfig/named file.

In the following sections, you'll experiment with the /etc/named.conf file. You should back it up in some fashion. Just be aware of the ownership and yes, the SELinux contexts of the file, as shown in this output:

```
# ls -Z /etc/named.conf
-rw-r-----. root named system_u:object_r:named_conf_t:s0 /etc/named.conf
```

If backups are restored haphazardly, even by the root user, the group ownership and /or the SELinux contexts may be lost. So if there's ever a failure in starting or restarting the named service, check the ownership and SELinux contexts of the /etc/named.conf file. If necessary, apply the following commands to that file:

```
# chgrp named /etc/named.conf
# chcon -u system_u -t named_conf_t /etc/named.conf
```

In addition, after testing a DNS configuration, some information may remain in a cache. That's the nature of a caching DNS server. If that cache still exists after a change to DNS configuration files, that could affect the results. Therefore, it's wise to flush the DNS cache after each configuration change with the following command:

```
# rndc flush
```

A Caching-Only Name Server

When you request a web page such as www.mcgraw-hill.com, the request is sent to the configured DNS server. The response is the associated IP address. The request is also known as a *name query*. For requests to external DNS servers, responses can take time. That's where a caching-only name server can help, as repeated requests are stored locally.

exam Watch

The default version of /etc/named.conf is set up for a caching-only nameserver, limited to the localhost system. Minor changes are required to open that server up to a local network.

When configuring a caching-only name server, the first step is to look at the default version of the /etc/named.conf configuration file. The directives in the default version of this file are organized to set up a caching-only nameserver. One view of this file is shown in Figure 17-1.

- The **options** directive encompasses several basic DNS directives, including the following:
  - The **listen-on port** (and **listen-on-v6 port**) directives specify the TCP/IP port number (for IPv4 and IPv6).

    To extend this to a local network, you'll need to include the IP address of the local network card. For example, for an IPv4 address of 192.168.122.50, you'd change the directive to read (don't forget the semicolon after each IP address)

    ```
    listen-on port 53 { 127.0.0.1; 192.168.122.50; }
    ```

    If IPv6 networking is active on the local network, you would need to configure similar IPv6 addresses for the **listen-on-v6** directive. If IPv6 networking is not active, the default **listen-on-v6** directive is sufficient.
  - The **directory** directive specifies where the DNS server looks for data files. Be aware, if the bind-chroot RPM is installed, these files are hard-linked to files in /var/named/chroot subdirectories.

FIGURE 17-1

/etc/named.conf for a caching-only nameserver

```
//
// named.conf
//
// Provided by Red Hat bind package to configure the ISC BIND named(8) DNS
// server as a caching only nameserver (as a localhost DNS resolver only).
//
// See /usr/share/doc/bind*/sample/ for example named configuration files.
//

options {
        listen-on port 53 { 127.0.0.1; };
        listen-on-v6 port 53 { ::1; };
        directory       "/var/named";
        dump-file       "/var/named/data/cache_dump.db";
        statistics-file "/var/named/data/named_stats.txt";
        memstatistics-file "/var/named/data/named_mem_stats.txt";
        allow-query     { localhost; };
        recursion yes;

        dnssec-enable yes;
        dnssec-validation yes;
        dnssec-lookaside auto;

        /* Path to ISC DLV key */
        bindkeys-file "/etc/named.iscdlv.key";
};

logging {
        channel default_debug {
                file "data/named.run";
                severity dynamic;
        };
};

zone "." IN {
        type hint;
        file "named.ca";
};

include "/etc/named.rfc1912.zones";
```

- The **dump-file** specifies the cache for the current DNS database and the output from the **rndc flush** command.
- The **statistics-file** specifies the cache for the current DNS database and the output from the **rndc stats** command.
- The **memstatistics-file** specifies the location for memory usage statistics.
- The **allow-query** lists the IP addresses allowed to get information from this server. By default, it's limited to the local system. To extend this to

another network such as 192.168.122.0/24, you'd change the directive to this:

```
allow-query { 127.0.0.1; 192.168.122.0/24; }
```

- Since BIND version 9.5, the software has included references to DNS security, in **dnssec-*** directives. The following directives enable DNS security, validation (to check authenticity), and querying, with the noted bindkeys-file:

```
dnssec-enable yes;
dnssec-validation yes;
dnssec-lookaside auto;
bindkeys-file "/etc/named.iscdlv.key";
```

- The **logging** directive specifies several more parameters; the **channel** directive specifies output methods, in this case to **default_debug**, activated in the named.run file in the /var/named/data, logging only **dynamic** issues.
- But wait, the **zone "."** directive specifies the root zone for the Internet, along with the root DNS servers as specified in the /var/named/named.ca file.
- Finally, the **include** directive includes the localhost settings described in the /etc/named.rfc1912.zones file.

No changes are required to create a caching DNS server, all you need to do is install the aforementioned **bind-*** packages, start the **named** service with the following command:

```
# /etc/init.d/named start
```

Next, run the **rndc status** command. If successful, you'll see output similar to that shown in Figure 17-2. The **rndc** command is the name server control utility.

Starting named

Make sure your computer is connected to a network. Now you can start the DNS server with the **/etc/init.d/named start** command. View the syslog message file with the **tail -f /var/log/messages** command. If there are problems, you'll see error messages in that file. As needed, the **named** daemon will display the file with the error. You can then stop the service with the **rndc stop** or **/etc/init.d/named stop** command and then check the applicable configuration files.

FIGURE 17-2

The status of an operational DNS server

```
[root@Maui michael]# rndc status
version: 9.7.0-P2-RedHat-9.7.0-5.P2.el6_0.1
CPUs found: 4
worker threads: 4
number of zones: 16
debug level: 0
xfers running: 0
xfers deferred: 0
soa queries in progress: 0
query logging is OFF
recursive clients: 0/0/1000
tcp clients: 0/100
server is up and running
[root@Maui michael]#
```

Once satisfied with the new configuration, make sure that DNS starts the next time you reboot Linux. As noted in other chapters, the following command makes sure that the **named** daemon starts the next time you boot Linux in the standard login runlevels:

```
# chkconfig named on
```

A Forwarding Name Server

This type of DNS server is simple. It requires a single command in the /etc/named.conf configuration file. As you can see, it's straightforward; I've set it to refer to a couple of other DNS servers on my home network:

```
options {
     listen-on port 53 { 127.0.0.1; };
     listen-on-v6 port 53 { ::1; }
     directory "/var/named";
     forward only;
     forwarders {
          192.168.122.1;
          192.168.0.1;
     };
};
```

With this configuration, any computer that looks to the local DNS server is forwarded to DNS servers on the IP addresses shown.

If you want to open up this server to external queries, a couple of more changes are required. The changes are the same as made earlier to the caching-only

nameserver configuration. In other words, for an IPv4 network, where the local network card has an address of 192.168.122.50, you'd change the **listen-on** directive to

```
listen-on port 53 { 127.0.0.1; 192.168.122.50; };
```

You should also include the **allow-query** directive described earlier, with references to the localhost system and the local network address:

```
allow-query     { localhost; 192.168.122.0/24; };
```

Forwarding from a Caching-Only Name Server

As suggested earlier, the caching-only name server configured in the default version of the /etc/named.conf file has forwarding features. Otherwise, it would not be able to return any results from DNS requests.

But you can combine aspects of the caching-only and forwarding name servers just described. Requests not in the local cache would be forwarded to the name servers specified with the **forwarders** directive. Figure 17-3 displays the relevant excerpt of a /etc/named.conf file where the forwarding directives have been included.

FIGURE 17-3

A caching nameserver that forwards to specific DNS servers

```
//
// named.conf
//
// Provided by Red Hat bind package to configure the ISC BIND named(8) DNS
// server as a caching only nameserver (as a localhost DNS resolver only).
//
// See /usr/share/doc/bind*/sample/ for example named configuration files.
//

options {
        listen-on port 53 { 127.0.0.1; 192.168.122.50; };
        listen-on-v6 port 53 { ::1; };
        directory       "/var/named";
     forwarders {
          192.168.122.1;
          192.168.0.1;
     };
        dump-file       "/var/named/data/cache_dump.db";
        statistics-file "/var/named/data/named_stats.txt";
        memstatistics-file "/var/named/data/named_mem_stats.txt";
        allow-query     { localhost; 192.168.122.0/24; };
        recursion yes;
```

BIND Troubleshooting Commands

There are four commands associated with the BIND service: **named**, **rndc**, **host**, and **dig**. The **named** command is the daemon in the /usr/sbin directory. It can be controlled by **named** script in the /etc/init.d directory. The preferred way to start DNS on a system is with the **/etc/init.d/named start** command. In contrast, the **dig** and **host** commands are successors to **nslookup**.

The **rndc** commands are straightforward. Try **rndc** by itself. The output guides you through the available options. The options I use are straightforward: **rndc status**, **rndc flush**, **rndc reload**, and **rndc stop**. If the DNS server is running correctly, the **rndc status** command should display the results shown back in Figure 17-2. The **rndc flush** command may help after changing a configuration file, as parts of the existing configuration may still reside in local memory. The **rndc reload** command rereads any changes made to the configuration or DNS database files. Finally, the **rndc stop** command stops the operation of the DNS server.

After you configure DNS and make it reread the configuration files with the **rndc reload** command, examine the results with the **host mheducation.com localhost** command. The output confirms the use of the local system as a DNS server and then provides a straightforward view of the IP address of the host and the hostname of the mail server:

```
Using domain server:
Name: localhost
Address: 127.0.0.1#53
Aliases:

mheducation.com has address 12.163.148.101
mheducation.com mail is handled by 0 mail.eppg.com.
```

Now test the setup. Use the **dig** command to examine your work. For example, with the command **dig @127.0.0.1 www.mcgraw-hill.com**, you'll see something like the output shown in Figure 17-4.

The dig command as shown asks the local DNS server to look for the www.mcgraw-hill.com server. Assuming IP address information for www.mcgraw-hill.com isn't stored locally, it then contacts one of the DNS systems listed in the named.conf file. If those systems are down or otherwise inaccessible, the local DNS server proceeds to forward the request to one of the name servers listed in the named.ca file. As though those are the root name servers for the Internet, the request will probably be passed on to other DNS servers. Therefore, it may take a number of seconds before you see an answer.

FIGURE 17-4

Test a local DNS server with the dig command

```
; <<>> DiG 9.7.0-P2-RedHat-9.7.0-5.P2.el6 <<>> @127.0.0.1 mcgraw-hill.com
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 24220
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 13, ADDITIONAL: 12

;; QUESTION SECTION:
;mcgraw-hill.com.               IN      A

;; ANSWER SECTION:
mcgraw-hill.com.        1660    IN      A       204.8.135.3
mcgraw-hill.com.        1660    IN      A       198.45.24.143

;; AUTHORITY SECTION:
.                       518195  IN      NS      e.root-servers.net.
.                       518195  IN      NS      j.root-servers.net.
.                       518195  IN      NS      a.root-servers.net.
.                       518195  IN      NS      h.root-servers.net.
:
```

EXERCISE 17-1

Setup Your Own DNS Server

Following the example files shown previously, set up one local caching DNS server. Access can be limited to the local system.

1. Install the bind package.
2. Review the contents of the /etc/named.conf file, based on the discussion so far in this chapter. Do not make any changes.
3. Start the DNS server with the following command:

   ```
   # /etc/init.d/named start
   ```

4. To make sure the DNS server is running, run the **rndc status** command. The output should be similar to that shown in Figure 17-2.
5. Flush the current cache with the **rndc flush** command.
6. Test the DNS server. Try the **dig @127.0.0.1 www.osborne.com** command.
7. If desired, you can now set up the local system as the DNS server. One method would be with the help of the network configuration tools described in Chapter 5.

CERTIFICATION OBJECTIVE 17.03

Set Up System Utilization Reports

As an administrator, it's helpful to know when a system is being overloaded. To that end, RHEL 6 includes the sysstat package. In addition, there are other commands related to measuring system utilization, specifically **top**. Of course, you can identify current disk usage with commands like **df** and **fdisk**. Once system utilization reports are collected, you can review the results, to help identify times when a system is in heavier use.

Of course, to paraphrase the relevant RHCE objective, there are other important commands that can help you "prepare and deliver reports" on the load on the CPU, RAM, hard drives, and the network. While they collect data similar to commands like **top**, **df**, and **fdisk**, the commands associated with the **sysstat** service collects such data on each of the noted components. It's configured to set up such data in log files. Then, the **sadf** command is designed to actually use that log data to prepare such reports. When written to an appropriate text or database file, such reports can then be delivered for evaluation and processing.

System Utilization Commands

Basic system utilization commands are already available for Linux. For example, the **top** command provides a current view of three important items: CPU, RAM, and processes. Take a look at the output of the **top** command, shown in Figure 17-5. Current CPU, RAM, and swap space use is shown atop the display; currently running processes are shown below the bar. Processes that take a lot of CPU and RAM are shown first. By default, the view is refreshed every three seconds.

Alternatively, there's the dstat command, part of the dstat package. As shown in Figure 17-6, it lists of a variety of statistics, refreshed every second. The one item added here relative to the top command is network traffic, which can help you view current network usage.

Of course, these are real-time statistics, and something that you can't stare at all the time. That's the reason behind the tools that I will call the system status service.

FIGURE 17-5

The top command displays system utilization.

```
top - 17:50:53 up  3:09,  2 users,  load average: 0.00, 0.00, 0.00
Tasks: 151 total,   1 running, 150 sleeping,   0 stopped,   0 zombie
Cpu(s):  0.0%us,  0.0%sy,  0.0%ni,100.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:    761264k total,   390932k used,   370332k free,    22188k buffers
Swap:  1023992k total,        0k used,  1023992k free,   142580k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 3523 root      20   0 14940 1180  872 R  0.3  0.2   0:00.04 top
    1 root      20   0 19244 1404 1132 S  0.0  0.2   0:00.67 init
    2 root      20   0     0    0    0 S  0.0  0.0   0:00.01 kthreadd
    3 root      RT   0     0    0    0 S  0.0  0.0   0:00.01 migration/0
    4 root      20   0     0    0    0 S  0.0  0.0   0:00.00 ksoftirqd/0
    5 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 watchdog/0
    6 root      RT   0     0    0    0 S  0.0  0.0   0:00.01 migration/1
    7 root      20   0     0    0    0 S  0.0  0.0   0:00.00 ksoftirqd/1
    8 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 watchdog/1
    9 root      20   0     0    0    0 S  0.0  0.0   0:00.02 events/0
   10 root      20   0     0    0    0 S  0.0  0.0   0:00.48 events/1
   11 root      20   0     0    0    0 S  0.0  0.0   0:00.00 cpuset
   12 root      20   0     0    0    0 S  0.0  0.0   0:00.00 khelper
   13 root      20   0     0    0    0 S  0.0  0.0   0:00.00 netns
```

The System Status Service

To set up the system status service, install the sysstat package. It's a regular server service, with its own daemon in the /etc/init.d directory. It's a cron job, run on a regular basis, as defined in the /etc/cron.d directory. It's a series of related commands, which are covered here.

FIGURE 17-6

The dstat command displays system utilization

```
[root@server1 etc]# man dstat
[root@server1 etc]# dstat
----total-cpu-usage---- -dsk/total- -net/total- ---paging-- ---system--
usr sys idl wai hiq siq| read  writ| recv  send|  in   out | int   csw
  0   0 100   0   0   0|  35k 2279B|   0     0 |   0     0 |  45    39
  0   0 100   0   0   0|   0     0 |  66B  114B|   0     0 |  42    35
  0   0 100   0   0   0|   0    16k|  66B  834B|   0     0 |  50    46
  0   0 100   0   0   0|   0     0 |  66B  354B|   0     0 |  39    31
  1   0 100   0   0   0|   0     0 |  66B  354B|   0     0 |  45    34
  0   0 100   0   0   0|   0     0 |  66B  354B|   0     0 |  46    34
  0   0 100   0   0   0|   0     0 |  66B  354B|   0     0 |  48    38
  0   0  97   3   0   0|   0    88k| 132B  468B|   0     0 |  49    45
  1   0  99   0   0   0|   0     0 |  66B  354B|   0     0 |  34    33
  0   0 100   0   0   0|   0     0 |  66B  354B|   0     0 |  47    36
  0   0 100   0   0   0|   0     0 |  66B  354B|   0     0 |  34    31
  0   0 100   0   0   0|   0     0 |  66B  354B|   0     0 |  35    33
  0   0 100   0   0   0|   0     0 |  66B  354B|   0     0 |  36    38
  0   0 100   0   0   0|   0     0 |  66B  354B|   0     0 |  38    36
  0   0 100   0   0   0|   0     0 |  66B  354B|   0     0 |  44    34
  0   0 100   0   0   0|   0     0 |  66B  354B|   0     0 |  35    31
  0   0 100   0   0   0|   0     0 |  66B  354B|   0     0 |  55    41
```

When the **sysstat** service is started, it uses parameters shown in the sysstat and sysstat.ioconf files, in the /etc/sysconfig directory. The sysstat file is relatively simple; the following directive specifies that log files should be kept for seven days:

```
HISTORY=7
```

And this directive specifies that log files that are more than ten days old should be compressed:

```
COMPRESSAFTER=10
```

Of course, that means system status log files are erased before they can be compressed. Naturally, you can change either variable as needed. The meaty /etc/sysconfig file is sysstat.ioconf, as it helps collect activity data from a variety of storage devices. It guides the **sysstat** service, to help it collect data from devices detected and listed in the /proc directory, in files named partitions and diskstats. While the sysstat.ioconf file is large, changes should not be required to that file unless there's new disk storage hardware. And the Red Hat exams are not hardware exams.

Collect System Status into Logs

The sysstat package includes a regular cron job. Available in the /etc/cron.d directory, that job collects information on system utilization and sends it to log files in the /var/log/sa directory. Examine the sysstat file in the /etc/cron.d directory. The first line suggests a job that's run every ten minutes, by the root administrative user.

```
*/10 * * * * root /usr/lib64/sa/sa1 -S DISK 1 1
```

The **sa1** command, with the 1 and 1 at the end, specifies that the job should be written once, one second after the job is started. The **-S DISK** option is associated with the **sar** command and collects utilization statistics on swap space. Information from this command is collected in file named sa*dd* in the /var/log/sa directory, where *dd* represents the day of the month.

The next line is more powerful than it looks. On a daily basis, at seven minutes before midnight, with the privileges of the root administrative user, the **sa2** command writes a daily report on just about everything to the noted file in the /var/log/sa directory.

```
53 23 * * * root /usr/lib64/sa/sa2 -A
```

As before, the **-A** switch is associated with the **sar** command. As suggested by the following excerpt from the **sar** man page, that essentially collects every reasonable bit on system utilization:

```
-A     This is equivalent to specifying -bBdqrRSuvwWy
-I SUM -I XALL -n ALL -u ALL -P ALL.
```

Information from this command is collected in file named sar*dd* in the /var/log/sa directory, where *dd* represents the day of the month. The sar*dd* files in that directory have already been processed into text files.

Prepare a System Status Report

This section will not prepare a report for a presentation. It's simply an analysis of the **sadf** command, and how it can be used to specify information to filter from the log files in the /var/log/sa directory. The binary log files with names like sa10 (for the tenth day of the month) can be processed in a number of ways by the **sadf** command. Some of the more important **sadf** switches are listed in Table 17-2.

For example, the following command sets up a report with data between the start and end of the tenth of the month:

```
# sadf -s 00:00:01 -e 23:59:59 /var/log/sa/sa10 > activity10
```

The data is redirected to the activity10 file, for later processing. But the power of the sysstat package comes from the way it interacts with other command, **sar**. But only some of the switches to the **sar** command work with **sadf**. As suggested in the **sadf** man page, the following command prepares a report based on "memory, swap space, and network statistics" from the /var/log/sa/sa21 file:

```
# sadf -d /var/log/sa/sa21 -- -r -n DEV
```

TABLE 17-2 Switches for the sadf Command

Switch	Description
-d	Display contents in a format usable by a database.
-D	Same as -d, except time is noted in number of seconds since 1/1/1970.
-e hh:mm:ss	List end time of report, in 24-hour format.
-p	Display contents in a format usable by the awk command; do not use with -d, -D, or -x.
-s hh:mm:ss	List start time of report, in 24-hour format.
-x	Display contents in XML format; do not use with -d, -D, or -p.

I cite the man page, as it's an excellent reference for the command required to create a report, while on the job, or even during a Red Hat exam. As with many other commands, this can be found in the EXAMPLES section. While the **-d** is associated with the **sadf** command, the double-dash (**--**) points to options associated with the **sar** command. So the **-r** reports memory usage, and the **-n DEV** reports statistics from network devices.

Of course, there are other important **sar** command switches. Those which may be relevant when you prepare a report on "processor, memory, disk, and network" utilization are described in Table 17-3.

With the switches listed in Table 17-3, you might modify the previous **sadf** command to meet all four of the items listed in the related RHCE objective:

```
# sadf -d /var/log/sa/sa21 -- -u -r -dp -n DEV
```

In other words, the **sadf** command specifies output usable by a database (**-d**) from the database file in the /var/log/sa directory associated with the twenty-first of the month. The double dash (**--**) points to **sar** command switches, with CPU utilization (**-u**); RAM utilization (**-r**); activity by block device; presented in more familiar block files such as sda (**-p**); with statistics from network devices (**-n DEV**).

TABLE 17-3

System Utilization Switches for the sar Command

Switch	Description
-d	Lists activity by block device, normally used with -p to specify common drive device filenames such as sda and sdb.
-n DEV	Reports statistics from network devices.
-P cpu	Lists statistics on a per-processor (or core) basis; e.g., -P cpu 0 specifies the first CPU.
-r	Reports memory utilization statistics.
-S	Collects swap space utilization statistics.
-u	Sets up a CPU utilization report, including categories related to applications executed at the user and system levels, idle time, and more.
-W	Reports swap statistics.

CERTIFICATION OBJECTIVE 17.04

Configure a System Logging Server

Briefly, while local logging is an RHCSA objective, remote logging is an RHCE objective. In other words, this section addresses the configuration of a system as a logging client, and a second system to receive associated data as a logging server. The rsyslog package, which should be installed by default even in minimal RHEL 6 configurations, represents the remote system logging server. With the proliferation of VMs, a central logging server is even more useful.

Chapter 9 describes the local logging aspects of the rsyslog package. If you're familiar with older Linux distributions, including RHEL 5, you should recognize that the local logging commands have not changed significantly. What has changed with the rsyslog package is the inclusion of modules, which can enable the transmission and receipt of log files over a given TCP/IP port; the default port for logs is 514.

The rsyslog package includes extensive documentation in HTML format, in the /usr/share/doc/rsyslog-4.6.2 directory (the version number may vary). If Internet access is not available in certain circumstances, such as during a Red Hat exam, critical files such as rsyslog_conf.html should be available in the noted directory. If you're familiar with the older syslog system, the rsyslog system should be easy. The configuration of a local log is identical for both systems.

exam Watch

In general, SELinux booleans do not directly affect log files. The SELinux booleans that affect individual services that may affect log messages are covered in other chapters.

System Logging Modules

Modules associated with the rsyslog service fall into three categories: input, output, and library. For the purposes of this chapter, you only need be concerned with input modules, and only those input modules already implemented or suggested in comments in the default /etc/rsyslog.conf file. Output modules are associated with sending data to other applications. Shortly, you'll examine those commands that send logging data from a client.

Enable Logging Clients

In this section, you'll look at those input modules listed in the standard rsyslog.conf configuration file. Some comments are already included in the default version of the file. The first command loads the imuxsock.so module with the **ModLoad** directive. The comment suggests that the module supports local system logging:

```
$ModLoad imuxsock.so    # provides support for local system
# logging (e.g. via logger command)
```

The next directive (**$ModLoad imklog.so**) loads the module associated with logging kernel-based information. The commented immark.so module, if active, would activate the **--MARK--** message periodically, if there is no other log activity.

The next two directives are important if you're configuring the system as a logging server. The module names (and associated comments) are descriptive, as imudp.so and imtcp.so refer to UDP and TCP communications, respectively.

```
# Provides UDP syslog reception
#$ModLoad imudp.so
# Provides TCP syslog reception
#$ModLoad imtcp.so
```

As noted in the commented directives described in the next section, UDP and TCP rsyslog communication is associated with port 514. For a logging server, the protocol you select depends on the importance of the information. UDP is associated with "best effort" delivery. While UDP is faster, it does not tell you if there's a transmission problem. But for many administrators, the occasional loss of data is trivial, given the large amounts of data often associated with log files.

If all of the data associated with log files are important, you should select the TCP module and directive. Of course, if you change the port number, you should also change the port number of any logging clients, as well as associated open firewall ports.

Configure Logging Servers

A logging server is a system set up to receive log information from clients. To review, such servers should have an open UDP or TCP port 514 to receive such information. On a real logging server, the disk space available on that server may be a concern. For example, some web servers create gigabytes of logging information every day. With that in mind, terabyte hard drives on logging servers may fill up quickly.

The default rsyslog.conf file includes suggested information for both UDP and TCP directives. Other directives can be added to further customize a logging server.

For more information, see www.rsyslog.com/doc/imudp.html and www.rsyslog.com /doc/imtcp.html.

UDP Directives

The standard suggested UDP directive specifies the port number where the server listens for logging clients:

```
$UDPServerRun 514
```

If there are multiple network cards on the local system, the following directive limits the logging server to the noted IP address:

```
$UDPServerAddress 192.168.122.50
```

TCP Directives

As with UDP, the standard suggested TCP directive specifies the port number where the server listens for logging clients. Multiple port numbers are supported, for configurations where you want to set up support for multiple clients.

```
$InputTCPServerRun 514
```

You can configure rsyslog to send a message if a logging client closes a connection:

```
$InputTCPServerNotifyOnConnectionClose on
```

If this server is working with a number of clients on different ports, you can use the **$InputTCPMaxListeners** directive to increase the number of such ports beyond the default of 20. This can work with the following directive, which sets the maximum number of supported sesssions (the default is 200):

```
$InputTCPMaxSessions
```

If you set a nondefault value of $InputTCPMaxSessions, that directive must be set before directives such as **$InputTCPServerRun 514**.

Configure Logging Clients

One of the features of the default rsyslog.conf file is the group of comments shown in Figure 17-7.

FIGURE 17-7

Draft logging client commands from /etc/rsyslog.conf

```
# ### begin forwarding rule ###
# The statement between the begin ... end define a SINGLE forwarding
# rule. They belong together, do NOT split them. If you create multiple
# forwarding rules, duplicate the whole block!
# Remote Logging (we use TCP for reliable delivery)
#
# An on-disk queue is created for this action. If the remote host is
# down, messages are spooled to disk and sent when it is up again.
#$WorkDirectory /var/spppl/rsyslog # where to place spool files
#$ActionQueueFileName fwdRule1 # unique name prefix for spool files
#$ActionQueueMaxDiskSpace 1g   # 1gb space limit (use as much as possible)
#$ActionQueueSaveOnShutdown on # save messages to disk on shutdown
#$ActionQueueType LinkedList   # run asynchronously
#$ActionResumeRetryCount -1    # infinite retries if host is down
# remote host is: name/ip:port, e.g. 192.168.0.1:514, port optional
#*.* @@remote-host:514
# ### end of the forwarding rule ###
```

As suggested in the comments, most of the directives are based on the use of TCP communication. All but the last commented directive define what happens on a logging client if it can't connect the network logging server.

The last commented directive applies to both TCP and UDP communications from a server. To configure the local system to send all logging messages, over TCP port 514, to a remote host on IP address 192.168.122.1, you'd add the following command:

```
*.* @@192.168.122.1:514
```

Alternatively, to configure just mail messages to be sent over UDP port 514 to a remote host on IP address 192.168.100.1, you'd add the following command:

```
mail.* @192.168.100.1:514
```

The *.* sends all logging messages. You can substitute options like authpriv, kern, and cron for the first asterisk. You can substitute log severity options such as debug, info, notice, warn, and so on for the second asterisk. A single @ represents UDP communication. A double @@ represents TCP communication.

Limit Access to Specified Systems

To limit access to a logging server, you'll need to configure an appropriate rule in an **iptables**-based firewall. For example, the following rule listed in the /etc/sysconfig/iptables file would allow communications from the noted IP address over TCP port 514.

```
-A INPUT -m state --state NEW -m tcp -p tcp -s 192.168.122.50 --dport 514 -j
ACCEPT
```

As noted in earlier chapters, if you use the Firewall Configuration tool, it's best to set this up as a custom rule. Otherwise, future changes in the Firewall Configuration tool may overwrite this custom rule.

CERTIFICATION OBJECTIVE 17.05

The Network Time Server Service

While NTP is no longer part of the RHCSA objectives, NTP as a client is implicitly covered in Red Hat's introductory RH124 course, a prep course for that exam. The configuration of NTP as a client is therefore covered in Chapter 5. In contrast, the configuration of NTP as a server is an RHCE objective covered here. NTP is classified as an RHCE-level network service. In other words, you need to know how to secure NTP just as you secure other network services such as Samba and NFS.

To allow NTP to work as a server, you need to allow access through UDP port 123. NTP uses the UDP protocol, because it's faster.

The NTP Server Configuration File

As discussed in Chapter 5, the configuration of an NTP client depends on the time zone documented in the /etc/sysconfig/clock file, as well as the servers configured in the /etc/ntp.conf file. Now it's time to configure one of those NTP servers, with the help of other settings in that ntp.conf file. Now you'll examine the default version of this file.

It starts with the **driftfile** directive, which monitors the error in the local system clock:

```
driftfile /var/lib/ntp/drift
```

There are also **restrict** directives that can help protect an NTP server. While a plain **restrict** directive works with IPv4 networking, a **restrict -6** directive works with IPv6 networking, Nevertheless, the same options are used with each **restrict** directive.

```
restrict default kod nomodify notrap nopeer noquery
restrict -6 default kod nomodify notrap nopeer noquery
```

The options to the **restrict** directive can be described as follows:

- **default** Refers to default connections from other systems; may be further limited by other **restrict** directives.
- **kod** Sends a "kiss of death" packet to systems that violate access restrictions; but it works only if there's a **limited** flag in the **restrict** line. So unless there's a security breach in the NTP server (always possible), the **kod** option can be removed.
- **nomodify** Denies queries that attempt to change the local NTP server.
- **notrap** Denies the control message trap service; you might remove this option to enable remote logging.
- **nopeer** Stops access from potential peer NTP servers. To configure a server that "synchronizes time using other NTP peers," as suggested in the RHCE objectives, you'd need to remove this option.
- **noquery** Ignores information and configuration requests.

But these restrictions, when combined, are good only for an NTP client. To set up an NTP server, specifically one that "synchronizes time using other NTP peers," you should remove at least the **nopeer** directive from this list. Some NTP servers may need to synchronize with yours, which is possible if you remove the noquery from the list as well.

The next two restrict directives limit access to the local NTP server to the local system. You should recognize the default IPv4 and IPv6 loopback addresses here:

```
restrict 127.0.0.1
restrict -6 ::1
```

Of course, when setting up an NTP server for other clients, you'll want to loosen that restriction. The comment that follows includes a network address in the required format. So to set up an NTP server for the 192.168.122.0/24 network, you'd change the **restrict** directive to

```
restrict 192.168.122.0 mask 255.255.255.0 notrap nomodify
```

For basic configuration, no additional changes should be required. Of course, the local NTP server should also be configured as a client to master NTP servers. And to repeat the reference from the RHCE objectives, the reference is to peers. The relevant directive is **peer**.

To test the directive on one NTP server, you could change the **server** in each of the default directives to **peer**, as shown here:

```
peer 0.rhel.pool.ntp.org
peer 1.rhel.pool.ntp.org
peer 2.rhel.pool.ntp.org
```

Alternatively, you could be given the URI to an NTP peer server, perhaps on a corporate network, perhaps on a network that's been configured during the exam.

Security Limits on NTP

As just described, the **restrict** directive from the /etc/ntp.conf configuration file can be used to limit access to a local NTP server. But that assumes an open port 123. Security limits can also refer to configured firewalls. Just be aware that an appropriate firewall rule for NTP opens up UDP (not TCP) port 123.

To test a connection to an NTP server, the **ntpq -p *hostname*** command can help. That command looks for the peers listed in the /etc/ntp.conf file. If the server is operational, you'll see something similar to the following output to the **ntpq -p localhost** command shown in Figure 17-8.

Of course, if that command works from a remote system, using the local hostname or IP address, you've verified that the remote NTP server is operational.

exam Watch

Since NTP is a UDP-based service, the* telnet *and* nmap *commands won't verify the operation of that service. You'll need a command like* ntpq -p *hostname for that purpose.

FIGURE 17-8

NTP server peers, verified with the ntpq -p command

```
[root@Maui michael]# ntpq -p localhost
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
 ntp.pbx.org     192.5.41.40      2 u   30   64   17  103.586  462.252  10.857
 ntp1.Housing.Be 169.229.128.214  3 u   22   64   17   38.452  462.901  12.111
 ntp.sunflower.c 132.236.56.250   3 u   23   64   17   74.502  536.253   5.584
[root@Maui michael]#
```

SCENARIO & SOLUTION	
You need to configure a caching-only DNS server for the local network	Use the default named.conf file; modify the **listen-on** and **allow-query** directives
You need to configure a caching-only DNS server to forward requests elsewhere	Use the named.conf file, modified for the caching-only directive; add a **forwarders** directive to point to a desired DNS server.
You need to set up a system utilization report for various hardware components	Start with the man page for the **sadf** command; use options associated with the **sar** command for desired components.
You want to configure a system logging server	Modify the rsyslog.conf file to activate a TCP or UDP module, and accept input over a port number.
You want to configure a system logging client	Modify the rsyslog.conf file. Use the commented options at the bottom of the file to point to a system logging server.
You need to set up an NTP server as a peer	Modify the ntp.conf file, to specify the host or IP address of a peer NTP server with the **peer** directive. Change the **restrict** directive of the server to remove the **nopeer** option and allow access to desired systems server.

CERTIFICATION SUMMARY

DNS provides a database of domain names and IP addresses that help web browsers, FTP clients, NTP clients, and more find sites on various networks, including the Internet. The default DNS server uses the **named** daemon, based on the Berkeley Internet Name Domain (BIND). It's a distributed database where each administrator is responsible for his or her own zone of authority, such as mcgraw-hill.com.

There are four basic types of DNS servers: master, slave (or secondary), caching-only, and forwarding-only. The RHCE objectives specifically exclude master and slave name servers. The default /etc/named.conf file is built in a caching-only DNS server configuration. A forwarding-only name server uses the **forward only** and **forwarders** directives in the named.conf file. In either case, you should configure the

listen-on and **allow-query** directives to support access from the local system and desired network(s). To test a DNS server, use commands like **rndc status**, **dig**, and **host**.

As an RHCE, you need to be able to monitor the performance of administered systems. That's the province of the sysstat service. While commands like **df**, **top**, and **dstat** can display CPU, RAM, disk, and network utilization data, actual reports can be prepared with the help of the sysstat service. That service is configured with the help of the sysstat and sysstat.ioconf files, with data collected on a periodic basis. Such data is collected into log files in the /var/log/sa directory with the help of a sysstat cron job in the /etc/cron.d directory. System status reports can be prepared from these log files with the help of the **sadf** command. An example of how this collects RAM and network data is available in the **sadf** man page; you can then add CPU and disk use data from related **sar** command switches.

With the proliferation of VMs, a central logging server can be more convenient. RHEL 6 includes the rsyslog package for that purpose. Administrators who are familiar with syslog should have no problem with rsyslog. Just be sure to activate TCP or UDP port 514 on the server, open up the associated port in the firewall, and add appropriate modules. The rsyslog.conf file even includes suggested directives for both logging clients and servers.

Finally, to configure an NTP server for a network, you need to do more to the associated configuration file, /etc/ntp.conf. As suggested in a file comment, the **restrict** directive should be changed to specify the network address. To support the peers suggested in the RHCE objectives, you also need a **restrict** directive without the **kod**, **noquery**, and (most important) **nopeer** options. Then to set up other systems as peers, you'd use the **peer *hostname*** format.

TWO-MINUTE DRILL

Here are some of the key points from the certification objectives in Chapter 17.

Basic Domain Service Organization

- DNS is based on the Berkeley Internet Name Domain (BIND), using the **named** daemon.
- Key packages include bind-chroot, which adds security by supporting DNS in a chroot jail, and bind-utils, which includes command utilities such as **dig** and **host**.
- Four basic types of DNS servers are: master, slave (secondary), caching-only, and forwarding-only. The RHCE objectives specifically exclude master and secondary DNS services.

Minimal DNS Server Configurations

- Critical DNS configuration files include /etc/sysconfig/named, /etc/named.conf and the files in the /var/named directory.
- The default /etc/named.conf is set up for a caching-only nameserver, limited to the local system. Changes to the **listen-on** and **allow-query** directives can enable access from DNS clients on a network.
- A forwarding name server requires a **forward only** or a **forwarders** directive that specifies the IP addresses of the remote DNS servers.
- The **forwarders** directive, with the IP address of the remote DNS servers, can be combined with the caching-only named.conf configuration.

Set Up System Utilization Reports

- System utilization reports are made possible in RHEL 6 with the help of the sysstat service.
- The sysstat service collects data regularly based on a job in the /etc/cron.d directory, in day-labeled files in the /var/log/sa directory.

- ❑ System status reports can be created with the **sadf** command, with an assist from **sar** command switches.
- ❑ A prime example of a system status report command is shown in the **sadf** man page, but it needs additional **sar** command switches to support the collection of CPU and disk performance information.

Configure a System Logging Server

- ❑ The RHEL 6 rsyslog service, for local clients, is essentially the same as the older syslog service.
- ❑ The rsyslog.conf file includes commented directives for the modules associated with the configuration of a system logging server.
- ❑ Default communications to a remote system logging server work over TCP or UDP port 514.
- ❑ Logging clients can be configured to send logs of all types and levels with a *.*, or the information that is sent can be customized. Information can be sent via TCP (@@) or UDP (@).

The Network Time Server Service

- ❑ The default NTP configuration file, /etc/ntp.conf, sets up a client with access limited to the local system.
- ❑ A standard **restrict** directive in the default ntp.conf file is available that opens access over UDP port 123 to systems on a specified network.
- ❑ The RHCE objectives suggest connections to peers; such connections can be configured by substituting **peer** for **server** In addition, you'd substitute the hostname or IP address of the remote peer servers for the default NTP servers such as 0.rhel.pool.ntp.org.
- ❑ Standard network test tools such as **nmap** and **telnet** don't work with the UDP communication associated with NTP networking.

SELF TEST

The following questions will help measure your understanding of the material presented in this chapter. As no multiple-choice questions appear on the Red Hat exams, no multiple-choice questions appear in this book. These questions exclusively test your understanding of the chapter. It is okay if you have another way of performing a task. Getting results, not memorizing trivia, is what counts on the Red Hat exams. There may be more than one answer to many of these questions.

Basic Domain Service Organization

1. What is the name of the RPM package that helps hide DNS files in a chroot jail? Do not include the version number.

   _____

2. Name two types of DNS servers are referenced in the RHCE objectives.

   _____

   _____

Minimal DNS Server Configurations

3. To configure DNS communication on port 53, what changes would you make to a firewall to support access by other clients to the local DNS server?

   _____

4. What file includes a basic template for a DNS caching-only nameserver?

   _____

5. What command can display the current operational status of the standard RHEL 6 DNS server?

   _____

6. What command makes sure that the DNS service starts the next time you boot Linux in a normal runlevel?

   _____

Set Up System Utilization Reports

7. What directory includes the job that creates standard system utilization reports? Assume the appropriate package is installed.

_____

8. On a RHEL 6 system, where can you find an example command to create a system utilization report? Where can you find additional switches for that report?

_____

Configure a System Logging Server

9. What module is associated with TCP reception for a system logging server?

_____

10. What command in /etc/rsyslog.conf sends all log information, using TCP, to a system on IP address 192.168.100.100, on port 514?

_____

The Network Time Server Service

11. Enter a directive, suitable for /etc/ntp.conf, that limits access to the 192.168.0.0/24 network. Hint: additional restrictions such as kod may be included in a different directive.

_____

12. What directive in /etc/ntp.conf is suited for connections to remote NTP peer servers?

_____

LAB QUESTIONS

Several of these labs involve configuration exercises. You should do these exercises on test machines only. It's assumed that you're running these exercises on virtual machines such as KVM. For this chapter, it's also assumed that you may be changing the configuration of a physical host system for such virtual machines.

Red Hat presents its exams electronically. For that reason, the labs in this and future chapters are available from the CD that accompanies the book, in the Chapter17/ subdirectory. In case you haven't yet set up RHEL 6 on a system, refer to Chapter 1 for installation instructions.

The answers for each lab follow the Self Test answers for the fill-in-the-blank questions.

SELF TEST ANSWERS

Basic Domain Name Service Organization

1. The package that supports additional security for a DNS name server based on BIND through a special chroot directory is bind-chroot.
2. Two types of DNS servers specified in the RHCE objectives are caching-only and forwarding.

Minimal DNS Server Configurations

3. To support access by other clients to the local DNS server, make sure TCP and UDP traffic is supported through the firewall on port 53.
4. The default /etc/named.conf file includes a basic template for a DNS caching nameserver.
5. Several answers work here, such as **/etc/init.d/bind status**, **rndc status**, and **service bind status**.
6. The command that makes sure that the DNS service starts the next time you boot Linux is

```
# chkconfig named on
```

   Similar commands such as **chkconfig named --level 35 on** are also acceptable answers.

Set Up System Utilization Reports

7. The directory with the standard sysstat job is /etc/cron.d.
8. On a RHEL 6 system, one place where you can find a command example of a system utilization report is the **sadf** man page. Additional switches can be found in the **sar** man page.

Configure a System Logging 2Server

9. The module associated with TCP reception for a system logging server is imtcp.so.
10. The command in /etc/rsyslog.conf that sends all log information, using TCP, to a system on IP address 192.168.100.100, on port 514 is

```
*.* @@@192.168.100.100:514
```

The Network Time Server Service

11. One directive in the /etc/ntp.conf file that limits access based on the noted conditions is

```
restrict 192.168.122.0 mask 255.255.255.0
```

    Of course, there are other ways to meet the same requirement. It's acceptable if the **restrict** directive includes options such as **kod**, **nomodify**, or **notrap**.

12. The directive in /etc/ntp.conf is suited for connections to remote NTP peer servers is

```
peer
```

LAB ANSWERS

Lab 1: Caching-only DNS Server

In this lab, you have the benefit of the /etc/named.conf configuration file. All you need to do is

1. Modify the **listen-on port 53** directive to include the local IP address; for example, if the local IP address is 192.168.122.150, the directive will look like

```
listen-on port 53 { 127.0.0.1; 192.168.122.150; };
```

2. Modify the **allow-query** directive to include the local IP network address:

```
allow-query { localhost; 192.168.122.0/24; };
```

3. Save your changes to /etc/named.conf.
4. Start the **named** service (other methods are available):

```
# service named start
```

5. Change the local client to point to the local DNS caching name server; replace any **nameserver** directives in /etc/resolv.conf with the IP address of the local system. For example, if the local computer is on 192.168.122.150, the directive is

```
nameserver 192.168.122.50
```

6. Test out the new local DNS server. Try commands such as **dig www.mheducation.com**. You should see the following near the end of the output:

```
;; SERVER: 192.168.122.50#53(192.168.122.50)
```

7. Point client systems points to the DNS server. Add the aforementioned **nameserver** directive to /etc/resolv.conf on those remote client systems:

```
nameserver 192.168.122.50
```

   To review the various methods to make the change permanent, see Chapter 3. One method is to set up **DNS1=192.168.122.50** in the configuration file for the network card, such as ifcfg-eth0 in the /etc/sysconfig/network-scripts directory.

8. To make sure the DNS service starts the next time Linux is booted, run the following command:

```
# chkconfig named on
```

9. Open up TCP and UDP ports 53 in the firewall on the local system. The simplest method is with the Firewall Configuration tool, which would include the following directives in the /etc/sysconfig/iptables file:

```
-A INPUT -m state --state NEW -m tcp -p tcp --dport 53 -j ACCEPT
-A INPUT -m state --state NEW -m udp -p udp --dport 53 -j ACCEPT
```

Lab 2: Caching and Forwarding DNS Server

As with Lab 1, the focus is on the configuration of the /etc/named.conf file. Nominally, the default caching-only DNS server already includes forwarding features, courtesy of the named.ca file listed near the end of the file. But to set up a specific forwarding server, you should add a forwarders entry such as the following in the **options** stanza.

```
forwarders { 192.168.122.1; };
```

Don't forget to configure a firewall, with an open UDP port 53. If necessary, you can set up host restrictions on the **iptables** rule that opens up that port, as discussed in the answer to Lab 1. You'll also need the rules to limit access to the local network. For example, the following output to the **iptables -L** command shows TCP and UDP port 53 (listed as domain in /etc/services) available to the 192.168.122.0/24 network:

```
ACCEPT  tcp  --  192.168.122.0/24  anywhere    state NEW udp dpt:domain
ACCEPT  tcp  --  192.168.122.0/24  anywhere    state NEW tcp dpt:domain
```

In general, it's best to set up such a change as a custom rule in the Firewall Configuration tool, if you want to use that tool in the future. As for the other requirements of the lab, you can clear the current cache with the **rndc flush** command, and reload the configuration file with the **rndc reload** command.

Lab 3: Create a System Utilization Report

If you understand this lab, the answer should be easy. While there are other methods, one appropriate command that meets the given requirements is available on the man page for the **sadf** command:

```
# sadf -d /var/log/sa/sa21 -- -r -n DEV
```

Of course, to get that information into the noted file, the output must be redirected:

```
# sadf -d /var/log/sa/sa21 -- -r -n DEV > sysstat_report.txt
```

Lab 4: Create a Detailed System Utilization Report

This lab builds upon what you did in Lab 3. If you haven't memorized the additional switches that specify information on CPU and disk usage, the appropriate switches are available in the man page for the **sar** command, for switches that apply after the double dash (**--**). As suggested in the man page, the **-u** can be used to report CPU usage, the **-d** reports activity by block device. It can help users read the output if the **-p** is combined with the **-d**.

But there's one more requirement: the **-p** next to the **sadf** command leads to output in a format usable by the **awk** command utility. The following way is one method to meet the requirements of the lab:

```
# sadf -p /var/log/sa/sa21 -- -u -r -dp -n DEV > morestat_report.txt
```

Lab 5: Configure a System Logging Server

Just a few changes are required to configure a system logging server. For example, to set up TCP communications, you can activate the following directives:

```
$ModLoad imtcp.so
$InputTCPServerRun 514
```

Of course, access requires an open port 514 in a local firewall. You could set up a custom rule with the Firewall Configuration tool such as the following:

```
-A INPUT -m state --state NEW -m tcp -p tcp --dport 514 -j ACCEPT
```

If you've selected UDP instead of TCP communication in the /etc/rsyslog.conf file, the associated custom firewall rule would change accordingly.

As a system logging server is not a regular network server in the RHCE objectives, this rule opens access to all systems. Of course, if needed, you can add such limits in a fashion similar to Lab 2. Don't forget to make sure the rsyslog service is running and set to start at appropriate runlevels upon reboot with a command like **chkconfig rsyslog on**.

Lab 6: Configure a System Logging Client

The default version of the /etc/rsyslog.conf file includes proposed directives that can be used to help configure a system as a logging client. While most of the directives aren't absolutely necessary, they can help save log data after a network break. So it's acceptable to activate one or more of these directives. Of course, for the final line, you'd want to substitute the IP address of the rsyslog server for the IP address shown here.

```
$WorkDirectory /var/spppl/rsyslog # where to place spool files
$ActionQueueFileName fwdRule1 # unique name prefix for spool files
$ActionQueueMaxDiskSpace 1g   # 1gb space limit (use as much as pos-
sible)
$ActionQueueSaveOnShutdown on # save messages to disk on shutdown
$ActionQueueType LinkedList   # run asynchronously
$ActionResumeRetryCount -1    # infinite retries if host is down
*.* @@192.168.122.50:514
```

Lab 7: Configure a NTP Server

In this lab, you'll set up one NTP server as a peer to another. That's possible with the **peer** directive, configured in the /etc/ntp.conf configuration file. For example, if a regular NTP server is configured on IP address 192.168.122.50, you can set up a peer on the 192.168.122.150 server with the following directive:

```
peer 192.168.122.50
```

Just remember, an NTP peer doesn't work unless the **nopeer** option has been removed from the **restrict** directive in the ntp.conf file.