Volume 7, Number 2 • February 2001
www.elementkjournals.com/sun
U.S. $11.00

Inside Solaris™
Tips & Techniques for users of Sun Solaris

Cut costs and clutter with headless servers

by Arthur Haigh

Operating system: Solaris

It has probably occurred to you that each of your servers doesn't need to have a monitor, keyboard and mouse, but for some reason we tend to configure them that way. The result is a cluttered server room and many wasted dollars.

The solution is to configure your servers as headless servers. A *headless server* is one configured without a monitor, keyboard or mouse.

In this article, we'll show you the options you have for reducing, if not eliminating, some of that clutter while improving your ability to remotely administer your servers. The solution can be as simple as a dumb terminal connected to your Sun box or as complex as a farm of network terminal servers.

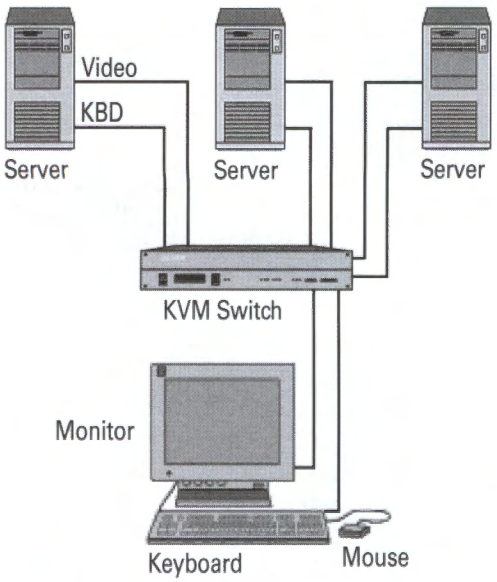

Figure A: *This is a keyboard, video and mouse (KVM) switch configuration.*

What's the system console?

The *system console* is the primary device over which system communication is performed. The choice of system console device is defined in the open boot prom (OBP) of the system. By default it's the monitor and keyboard.

The system console displays error and warning messages for the system as it's configured to do in the /etc/syslogd.conf file. While the system is booting, it displays the Power-On Self-Test (POST) results as well as all of the standard boot messages to the system console. Should the system fail to boot, the administrator is prompted via the system console. It's only via the keyboard of the system console that you can issue the abort keystroke [STOP]A to halt the system.

The console configuration

The system console device is specified by two parameters in the OBP. By default, the NVRAM is configured to address a keyboard as the standard input device to the server. The standard output device is

In This Issue

element k JOURNALS
A Division of Element K Press

configured as the monitor. You can verify the configuration of your server by looking at the OBP environment variables `input-device` and `output-device`. You can do this two ways; one of which is from the `ok` prompt as seen here:

```
ok printenv  input-device
Input-device = keyboard
ok printenv  output-device
output device=monitor
```

You can also use the `eeprom` command directly from Solaris. This command is platform-specific and, therefore, the platform type is part of the path. You may substitute the platform in the following path or simply allow the output from the `uname` command to substitute it in for you. The advantage of the form given below is that it will work on all platform types:

```
# /usr/platform/`uname -i`/sbin/eeprom
  ➥input-device
input-device=keyboard
# /usr/platform/`uname -i`/sbin/eeprom
  ➥output-device
output-device=screen
```

The banner display, shown during system boot, will show you if it detects a keyboard connected to the system. You should see a message similar to the following:

```
Type-5 keyboard present.
```

Reducing clutter while maintaining your GUI

One solution to the overcrowded server room is the elimination of keyboards and monitors by utilizing a keyboard, video and mouse console switch, also known as a KVM switch. This device allows you to attach multiple servers to a single keyboard, mouse and monitor. The switches are quite convenient and allow you to switch among servers via a touch pad or front panel switch. A typical arrangement is shown in **Figure A** on the cover.

Switching doesn't affect the state of the system, and you can switch active sessions. A typical switch might allow for the control of five servers. You can gather the switches together so that two switches could centrally control nine servers (one input group is lost in wiring the switches together). This option provides you the same graphical user interface that a dedicated keyboard, mouse and monitor (KMM) provides. You may need to purchase longer video and keyboard cables depending on the physical position of the switch and servers.

Using of the KVM will save you money if you don't have to purchase monitors and keyboards for servers being connected to the KVM. The video adapter card is still required. A good KVM switch costs up to $2,000, so this option may very well prove more costly to you should you need to purchase video cable and/or adapters for monitors you already own.

Configuring a KVM console switch

If you wish to reconfigure your servers with a KVM console switch, then the `eeprom` settings given earlier are correct and needn't be modified. You'll need to schedule a system reboot to move the keyboard cables. The monitor cables can be moved at your convenience, but until both keyboard and video cables are connected to the KVM switch, the system console will be disrupted.

Many of the mid- to high-range Sun Servers (such as the E3500) have key-switched locks that enable you to limit keyboard access to the system while the key switch is in the locked position. This prevents halting the system via the [STOP]A keystroke. It doesn't, however, prevent the system from crashing if you unplug and plug in the keyboard cable. In fact, the behavior of the system is rather strange with respect to the effects of unplugging or plugging in the keyboard.

Replacing the keyboard, mouse and monitor with a dumb terminal

As you may have noticed in the description of the system console, there's no mention of a graphical interface. There's no requirement that the system console have graphic capabilities. As such, we feel there's no reason for using a keyboard, mouse and monitor as the system console. Graphical capabilities aren't lost, however, as you can initiate a remote CDE or an open Windows session via XDCMP using an X client, such as Reflection X running on a PC, or the CDE remote host access option from a UNIX system running CDE.

While we can eliminate the presence of the keyboard, mouse and monitor, we still require access to the system console for system interaction during the boot sequence and in the event of a system failure. For this access, we can configure a simple character-based terminal. Because these terminals (such as the DEC VT-220 or Wyse terminal) don't possess any CPU intelligence, they're known as dumb terminals.

A *dumb terminal*, unlike a standard keyboard, can be unplugged from the server without chang-

ing the state of the server, because the keyboard port on the server isn't used. The terminal connects to the server via an inexpensive serial cable wired in accordance with the Institute of Electrical and Electronics Engineers (IEEE) RS-232 standard. The RS-232 cable connects to one of the standard serial ports on the server typically called TTYA or TTYB.

Configuring a dumb terminal as the system console

Two sets of parameters need to be checked and, perhaps, modified to configure a dumb terminal as the system console. These sets are first the OBP NVRAM environment variables `input-device` and `output-device` discussed previously, and second the communication parameters of the terminal and server. As in the examples we've shown, you can view and modify these parameters via the OBP or the OS. A system reboot is required, in any case, to implement any changes you've made. If you make a mistake while configuring these parameters, you may be forced to reset all of your OBP NVRAM settings by issuing the [STOP]N keystroke during boot from a power up. Be careful when issuing the [STOP]N keystroke during boot, as you could inadvertently delete NVRAM aliases or other settings that may be needed for booting.

The communications parameters are most easily set on your terminal. You can view the configuration of the communication parameters on your server and then change the terminal to match. To display them on the server via the OBP, use the following:

```
ok  printenv ttya-mode
ttya-mode=9600,8,n,1,-
```

Or via the OS using the `eeprom` command like this:

```
# /usr/platform/`uname -i`/sbin/eeprom
  ➥ttya-mode
ttya-mode=9600,8,n,1,-
```

The fields are baud rate (9600), number of data bits (8), parity (none), stop bits (1), and handshake (- indicates none). The values shown are the default values.

Now that the communication settings are configured on the terminal, we can proceed and configure the server to recognize the terminal as the system console. If you're using the OBP, issue the following commands:

```
ok setenv input-device=ttya
Input-device=ttya
ok output-device=ttya
```

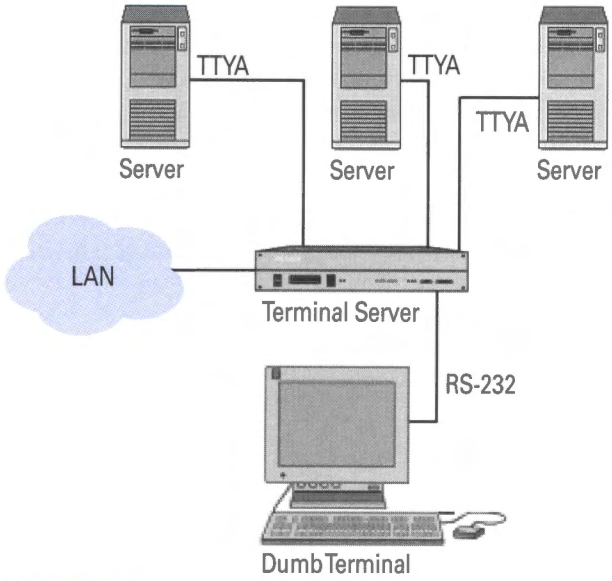

Figure B: *This is a typical terminal server configuration.*

```
Output-device=ttya
ok reset
```

Or, via the OS, you'll use these commands:

```
# /usr/platform/`uname -i`/sbin/eeprom
  ➥input-device ttya
input-device=ttya
# /usr/platform/`uname -i`/sbin/eeprom
  ➥output-device ttya
output-device=ttya
# reboot
```

This completes the configuration for using your terminal. The systems will now reboot and use the terminal as the system console.

Adding remote access

Consolidating the system consoles to a dumb terminal is a significant simplification. Unfortunately, you still need to be sitting in front of the terminal to access the system consoles. A terminal server is a device created to solve this problem.

A *terminal server* is a network device that allows remote access to the serial lines on the servers you have configured for dumb terminal access. While the terminal switch allows you to access one of many ports via a single dumb terminal, the terminal server allows you to access remotely, via telnet, one of many serial ports. **Figure B** depicts the wiring of a typical terminal server.

You can connect the terminal server to a LAN and also, via one of the serial ports, to a modem as well as a dumb terminal. The configuration of the servers is the same as for wiring up the dumb terminal and/or the terminal switch. Wiring the

terminal server to a LAN or modem opens potential security issues that you need to consider seriously before implementing such a configuration.

Terminal caveats

There's a bug with versions of Solaris earlier than Solaris 8. If you have the system console configured for a dumb terminal or terminal server, and you lose power to the terminal (or simply turn the terminal off), it will change the voltage levels on the RS-232 port, causing the server to interpret this as a BREAK sequence forcing the OS to halt. Sun has issued a patch for Solaris 2.6 (patch # 105924) and Solaris 7 (patch # 107589), and these are described in Infodoc 21258. The bug is fixed in Solaris 8. Unfortunately, if you're running even older versions of Solaris, you must purchase a fix from Sun Professional services called ZSBRK (see Sun Infodoc 18666).

Headless helpers

You can reduce cost, decrease clutter, and improve the remote accessibility of your servers by configuring them to be headless servers. It isn't difficult to do and has proven to be well worth the effort. ✳

Getting value from Solaris core files

by Rob Thomas

Operating system: Solaris 7/8

Core files are historically misunderstood and often deleted pieces of data on the disk. Many system administrators create cron-based scripts to scrub the system of all core files. However, core files contain critical information that will assist the debugger in determining the cause of the crash. In this article, we'll discuss core files as well as the management and creation of core files through the Solaris 7 and 8 commands `coreadm(1M)` and `gcore(1)`.

A bit about core files

Core files are created when an application receives certain signals. The name comes from the distant past, when main memory was called "core." These files contain critical bits of data from the application, to include register contents, status and data from memory. Further, these files can prove highly valuable to an analyst when investigating the cause of a crash. You can create a core file from any number of applications, as well as the Solaris kernel itself.

Core files are popularly labeled as needless. However, this isn't at all the case. A core file contains valuable system state information that the programmer requires to determine the cause of the mishap. By using debuggers such as `dbx` or `gdb`, the programmer can trace the last steps of the application and unearth the root cause of the application failure.

Core files are also caused by system crashes. The same rules apply here—the crash dumps are critical to the crash analyst's review of the failure. These crash dumps often contain the smoking gun that reveals the cause of woe, be it hardware- or software-based. These files will be created after a crashed system has rebooted, and are managed by the `savecore(1M)` command. By default, Solaris 7 enables the `savecore` command in /etc/init.d/savecore.

If you're deleting all core files through a crontab entry or something similar, you may be removing all evidence of a misbehaving application. Worse, when this application begins to go awry in more visible ways, the application support personnel may be unable to debug the problem. For this reason, core files should be analyzed prior to deletion.

Whodunit?

Determining the creator (both application and user) of a core file can be a tricky task. By default, the application will create the core file, named *core*, in its working directory. System crashes are saved under /var/crash/`uname –n.

Application core dumps will overwrite the previous core file, leaving the system administrator to guess how often the application is dumping its core. This certainly makes problem determination difficult. Further, if the directory permissions are restrictive to the UID, the core may not be dumped at all.

Each system crash will create two files, vmcore and unix. The *vmcore* file contains the data resi-

dent in memory at the time of the crash. The *unix* file contains the name list associated with the kernel running at the time of the crash. To avoid overwriting previous crash dumps, the system appends a number to each file, incrementing this number with each crash. The next number to be used is contained in the /var/crash/`uname–n`/bounds file.

Fortunately, Solaris 7 and 8 provide a tool to manage core dumps more effectively: `coreadm`. Let's take a look at how this tool can help us.

Using coreadm(1M)

The `coreadm` tool provides the system administrator with an interface for managing the parameters that affect core file creation. The `coreadm` command modifies the /etc/coreadm.conf file. This file is read at boot time and sets the global parameters for core dump creation.

With `coreadm`, we can modify the core filenaming parameters to provide greater detail and some degree of unique naming. Here's the default output from `coreadm`:

```
root@bilbo# coreadm
     global core file pattern:
       init core file pattern: core
            global core dumps: disabled
      per-process core dumps: enabled
     global setid core dumps: disabled
per-process setid core dumps: disabled
     global core dump logging: disabled
```

Note that the global core file pattern is null. By modifying this field, we can create more specific core filenames. First, let's set our core filenames to be of the form core.<PROC NAME>.<PID>. We'll do this only for all programs we execute in this shell (the `$$` notation equates to the PID of our current shell):

```
: bilbo; coreadm -p core.%f.%p $$
```

The `%f` indicates that the program name will be included, and the `%p` indicates that the PID will be appended to the core filename. We'll use the following program, `makecore.c`, to generate core dumps, which uses the `assert()` macro to create the dump files:

```
/*
 * @(#)makecore.c 20 AUG 2000
 * Rob Thomas robt@cymru.com
 * A program that always dumps core.
 */

#include <assert.h>

int
main(int argc, char **argv) {

   assert(1 == 2);
}
```

Now when we execute the `makecore.c` program, our core dump filename is a bit more descriptive, as noted below:

```
: bilbo; ./makecore
dump.c:11: failed assertion `1 == 2'
Abort(coredump)
: bilbo; ls -l core*
-rw-------   1 robt     mopes
➥66184 Aug 20 19:06 core.makecore.5565
```

We now know that the `makecore` program created the core file, and that `makecore` had a PID of 5565. With each successive execution of the dump program, we generate a new, uniquely named core file.

Perhaps, however, we're dumping core on a shared file system that's NFS mounted on several workstations. Which workstation has the problem? The `coreadm` command can assist us here as well. We'll utilize the `%n` flag to accomplish this task:

```
: bilbo; coreadm -p core.%n.%f.%p $$
: bilbo; ./makecore
makecore.c:12: failed assertion `1 == 2'
Abort(coredump)
: bilbo; ls -l core*
-rw-------   1 robt     mopes
➥66184 Aug 22 14:27 core.bilbo.makecore.10267
```

Now we can determine the process name, the process ID, and the hostname that created the core file.

Managing core files

The next task is to better manage the creation and storage of core files. In this scenario, we'll place all core files in a new directory, /var/adm/corefiles. Further, we'll track the process ID and process name that created the core file. This will be a global setting for all users, and thus we must be root to perform this task. Ensure that you have sufficient disk space in the partition that contains the /var/adm/corefiles directory! We create the directory as follows:

```
root@bilbo# mkdir /var/adm/corefiles &&
➥chmod 1777 /var/adm/corefiles
```

We create this directory as mode 1777 (owner, group, other can read, write, and execute, but not

overwrite, each other's files) to ensure that anyone may dump core into this directory. Keep in mind that the kernel creates core files under the UID of the owner of the process, and further protects the core file by creating it in mode 600. This will prevent prying eyes from viewing the address space of other users. Now let's set our global `coreadm` features. The following will change the core file patterns globally and for init, and will require a reboot to take effect:

```
root@bilbo# coreadm -g
➥/var/adm/corefiles/core.%f.%p -i
➥/var/adm/corefiles/core.%f.%p
```

Once the system is rebooted, all future core files will be placed in the /var/adm/corefiles subdirectory. You can verify these settings by reviewing the /etc/coreadm.conf file. Don't edit this file directly; rather, use the `coreadm` command to modify it.

Creating core files from live processes

It may become necessary to force a core dump from a running process. It may also be the case that you can't afford to actually halt the process. As previously noted, core files yield a great deal of information regarding the state and behavior of a process. The ability to determine the state of a process can be quite helpful when attempting to determine why the process has gone awry. Fortunately, Solaris provides the `gcore(1)` command for just this purpose.

When you use `gcore` on a running process, the kernel creates a core file of the form core <PID> in the current directory. This core file can then be sent off to the vendor or support personnel for review. You can change the name of the core file with the `-o` option. Here's a sample:

```
: spanky; pgrep linger
265
: spanky; gcore 265
gcore: core.265 dumped
: spanky; ls -l
total 190
-rw-r--r--   1 robt     mopes
➥66176 Aug 22 16:25 core.265
```

The `linger` program is simply a home-brew program that sleeps for several hundred seconds. The `pgrep` command returns the PID of the linger program (Solaris 8 only; otherwise use `ps -ef | grep linger`), which we use as an argument to `gcore`. The core file, `core.265`, is created and can now be analyzed to help uncover the cause of any issues.

Wrapping up

While core files are often believed to be nothing more than a waste of disk space, they can actually be quite helpful in determining the cause of an issue with a given application. Further, it's possible to create a core file from a running application, possibly yielding the necessary information to solve some thorny application issues. The `coreadm` and `gcore` commands are powerful additions to any Solaris administrator's toolkit. ✳

Understanding Internet protocols: TCP and UDP

by Edgar Danielyan

Operating system: Solaris

The suite of network protocols used to power the Internet is called TCP/IP. As you can see, even the name of this suite shows the importance of TCP (Transmission Control Protocol). And while the User Datagram Protocol (UDP) isn't honored with such a mention, it's nevertheless equally important. In this article, we'll describe both of these transport protocols at the fourth layer of the OSI Model. It's estimated that more than 90 percent of Internet traffic is carried either in TCP or UDP packets, so these protocols deserve a detailed overview.

TCP

TCP provides a connection-oriented, ordered, full-duplex-guaranteed delivery transport in IP networks. TCP is an intelligent protocol, hence its complexity and the overhead associated with its

use. TCP has evolved over time, and the Internet community has made many improvements and additions to the basic protocol.

TCP header format

TCP is one of the most complex network protocols, and its packet header reflects it. As you can see, it takes a lot to provide a connection-oriented and confirmed-delivery transport over a connectionless, best-effort delivery IP network. **Table A** describes the contents of the TCP header and each of its fields' functions.

TCP operation

TCP operation has three phases: handshake, data transmission, and connection closing. Let's start with the description of handshake.

Handshake

Connection establishment, also called handshake, is necessary before any data can be sent or received—this is true for any connection-oriented protocol such as TCP. TCP uses a *three-way handshake* procedure, which works as follows:

- The initiator of the connection sends a TCP packet with the SYN flag and an initial sequence number set.
- The remote system receives that TCP packet and responds with a packet where ACK is set and that contains its initial sequence number.
- The initiator responds with a packet that has the ACK flag set and with the initial sequence number of the remote end.

When these three operations are complete, a TCP state machine is created at both ends of the connection and the TCP connection is established. After you've established the connection, data may be exchanged.

Slow start

After the TCP connection is established, TCP starts by sending a small chunk of data to see the condition of the network. If these initial packets do well, it starts increasing the rate at which it sends packets until the network shows signs of congestion (e.g., slow delivery and/or packet drops). TCP tries to achieve the optimal transmission speed using *sliding windows*, where the general principle of flow control is based on the proactive management of windows and timeouts. The receiving endpoint tells the sending endpoint that it has so much available buffer space. The sender transmits up to this amount of data before waiting for acknowledgment from the receiving endpoint.

Since TCP is an adaptive protocol, it tries to adapt to the network conditions at any particular point in time, and doesn't offer much predictability. This is why it's not usually used for isochronous (timing dependent) transmission, such as Voice over IP (VoIP). TCP, from the moment of

Table A: *TCP packet headers and fields*

Field	Description
Source port (16 bits) Destination port (16 bits)	Distinguishes particular connections and protocols.
Sequence number (32 bits)	Works to properly reassemble the data stream.
Acknowledgment number (32 bits)	Informs the sending endpoint of successful receipt of previously received data.
Data offset (4 bits)	Contains the number of 32-bit words in the TCP header.
Flags (URG, ACK, PSH, RST, SYN, FIN) (6 bits)	Specifies the type and purpose of the packet. URG is set if the Urgent Pointer is applicable. ACK shows whether this is an acknowledgment. PSH is set to 1 if the sender wants the recipient to push the data in the packet. If RST is set, then this resets the TCP connection. SYN is used in connection establishment and synchronizes two endpoints. Finally, FIN closes the connection.
Window (16 bits)	Indicates the size of available buffers.
Checksum (16 bits)	Represents the checksum of the entire TCP header.
Urgent pointer (16 bits)	Used only if the URG flag is set.
Options (24 bits)	Indicates TCP options, if used.

connection establishment, tries to increase the dataflow rate to maximally utilize the available bandwidth. Then as saturation and packet loss begin to show, it reduces the speed at which it sends packets.

Since conditions in the network may change often, and on the Internet they change constantly, this water-testing operation may be performed many times over the lifetime of a TCP connection. When, for example, a second TCP connection is established over the same link, the two connections soon will use approximately half of the link's bandwidth. When the third and fourth connections come up, the bandwidth will be divided even further. However, it's necessary to keep in mind that, besides TCP, there are other protocols traveling across the network, so that these calculations in fact are quite approximate.

In performing its functions, TCP has to be able to detect and deal with packet loss, which is most often the result of network congestion, when routers have to drop packets. TCP contains complex mechanisms that detect loss of packets and arrange for retransmission.

Connection closing

When the connection has to be closed, the initiator sends a packet with the FIN flag set. However, the TCP state machine may also close a connection. This can happen either from inactivity (i.e., timeout) or for other reasons.

Protocols using TCP

The following application-level protocols use TCP (in no particular order): Telnet, FTP, SMTP, POP, SSH, HTTP, HTTPS, BGP, NNTP, X11, IMAP, whois, finger, and others. As you can see, a large portion of network traffic is TCP-based.

User Datagram Protocol

UDP (IP protocol number 17) is described in RFC 768 by the late Dr. Jon Postel. The major traditional users of UDP are the Domain Name System (DNS) and the Trivial File Transfer Protocol (TFTP). More modern uses of UDP include audio and video over IP, including, in particular, IP Telephony applications. UDP is used in these areas because of its low protocol overhead and flexibility characteristics.

UDP header format

As you can see, UDP headers are incomparably simpler than TCP, or even IP, headers:

- Source port (16 bits)
- Destination port (16 bits)
- Length of the packet (16 bits)
- Checksum (16 bits)

There are no notions of handshake or connection synchronization because UDP is a connectionless protocol. There's no bandwidth utilization management or anything similar, as found in TCP. ✳

References

- User Datagram Protocol (RFC 768)
- Transmission Control Protocol (RFC 793)

Supporting Active Directory with BIND

by Don Kuenz

Application: BIND 8.1.2 or later
Operating systems: Solaris, Windows 2000

The introduction of Active Directory (AD) into Windows 2000 imposes new requirements on Domain Name Services (DNS). In this article, we'll show you how to support AD from the Berkeley Internet Name Domain (BIND) system, which Solaris uses for DNS.

AD requires a recent version of BIND that can support service location (SRV) resource records. You also need to configure BIND to allow underscores within SRV records. You can optionally choose to enable dynamic updates to save yourself the trouble of manually entering SRV records.

We begin by explaining how to display BIND's version using `nslookup`. Next, we show you how to configure /etc/named. Finally, we tell you how to test your configuration.

This article uses a DNS server named apollo as a name server for the biz.com domain, which uses IP addresses in the 192.168.1.1 to 192.168.1.255 range. We also use a host named aphrodite as a Windows 2000 server. You need to use your own host names, domain name, and IP address range to correctly set up your own DNS server that supports AD.

AD requirements

BIND version 8.1.2 or later can support AD. Use the following `nslookup` commands to check the version of BIND running on your host:

```
D:\>nslookup
Default Server:  apollo.biz.com
Address:  192.168.1.1

> set class=chaos
> set type =txt
Unrecognized command: set type =txt
> set type=txt
> version.bind
Server:  apollo.biz.com
Address:  192.168.1.1

VERSION.BIND    text =

     "8.2.2-P5"
```

If you discover that you currently use an older version, you need to download and install the latest version of BIND from **www.isc.org /products/BIND/**.

Listing A shows some of the resource records that AD uses. The CNAME resource provides an alias for a long string of numbers to a host name. Microsoft calls such strings of numbers Global Unique Identifiers (GUIDs).

Listing A also shows SRV resource records. Although BIND version 8.1.2 and later can support SRV records, AD's SRV records typically start with an underscore. By default, BIND disallows underscores. In the next section, we show you how to tell BIND to ignore the underscores.

The AD functionality within each Domain Controller (DC) generates all of the records shown in **Listing A** for each DC within your network. You could manually enter these records, but you may find it easier to allow BIND to automatically add records by enabling dynamic updates. This saves a lot of tedious work, particularly when you deploy multiple DCs.

Listing A: *A partial dump of /var/named/db.biz.com that contains CNAME and SRV resource records used by Active Directory*

```
$ORIGIN _msdcs.biz.com.
84bf3269-def3-4bca-8da7-7d372067c97a 600 IN CNAME
➥aphrodite.biz.com. ;Cl=2
ca70fc3e-5845-4964-94c3-cb59801709fe 600 IN CNAME
➥aphrodite.biz.com. ;Cl=2
fe9d89be-6d46-4491-99b9-50a9affc5c10 600 IN CNAME
➥aphrodite.biz.com. ;Cl=2
$ORIGIN _tcp.dc._msdcs.biz.com.
_ldap     600 IN SRV 0 100 389 aphrodite.biz.com. ;Cl=2
_kerberos 600 IN SRV 0 100  88 aphrodite.biz.com. ;Cl=2
$ORIGIN _tcp.Default-First-Site-Name._sites.dc._msdcs.biz.com.
_ldap     600 IN SRV 0 100 389 aphrodite.biz.com. ;Cl=2
_kerberos 600 IN SRV 0 100  88 aphrodite.biz.com. ;Cl=2
$ORIGIN _tcp.pdc._msdcs.biz.com.
_ldap     600 IN SRV 0 100 389 aphrodite.biz.com. ;Cl=2
$ORIGIN _tcp.ee7d7ca7-8b95-4815-abdf-c345bf1d8b68.
➥domains._msdcs.biz.com.
_ldap     600 IN SRV 0 100 389 aphrodite.biz.com. ;Cl=2
$ORIGIN _tcp.d6e8983b-fb2c-4905-b389-44816003b88b.
➥domains._msdcs.biz.com.
_ldap     600 IN SRV 0 100 389 aphrodite.biz.com. ;Cl=2
$ORIGIN _tcp.f598a41e-fcb9-42c7-a86e-ed3c9e4c769f.
➥domains._msdcs.biz.com.
_ldap     600 IN SRV 0 100 389 aphrodite.biz.com. ;Cl=2
$ORIGIN _udp.biz.com.
_kpasswd  600 IN SRV 0 100 464 aphrodite.biz.com. ;Cl=2
_kerberos 600 IN SRV 0 100  88 aphrodite.biz.com. ;Cl=2
$ORIGIN _tcp.Default-First-Site-Name._sites.biz.com.
_ldap     600 IN SRV 0 100 389 aphrodite.biz.com. ;Cl=2
_kerberos 600 IN SRV 0 100  88 aphrodite.biz.com. ;Cl=2
$ORIGIN _tcp.biz.com.
_ldap     600 IN SRV 0 100 389 aphrodite.biz.com. ;Cl=2
_kpasswd  600 IN SRV 0 100 464 aphrodite.biz.com. ;Cl=2
_kerberos 600 IN SRV 0 100  88 aphrodite.biz.com. ;Cl=2
```

Configuring BIND

We only need to make changes to BIND's primary configuration file to enable AD support. BIND names its primary configuration file /etc/named .conf by default. **Listing B** on the next page shows how /etc/named.conf looks for a domain named biz.com that uses IP addresses in the 192.168.1.1 to 192.168.1.255 range.

Look at the two zones named `biz.com` and `192.168.1.in-addr.arpa` in **Listing B**. Both of those zones contain the following three lines:

```
allow-transfer{ 192.168.1/24 };
allow-update{ 192.168.1/24; };
check-names ignore;
```

Notice that the first two lines use an *address match list* to specify a range of IP addresses. Our

address match list includes all IP addresses whose first 24 bits contain 192.168.1. This specifies the IP address range from 192.168.1.1 to 192.168.1.255.

Our `allow-transfer` statement specifies the IP addresses of hosts that can receive zone transfers from our server. The `allow-update` statement specifies the IP addresses of hosts allowed to add, modify or delete records or RR sets in master zones. The `check-names` statement disables name checking, which permits Windows 2000 to use underscores in the names that appear within the zone.

Testing your configuration

You can test your configuration by either installing Windows 2000 as a DC or by booting a Windows 2000 DC. Use Windows Event Viewer to see if any AD errors appear in the System Log.

Listing B: *The contents of the etc/named.conf file*

```
options {
   ///etc/named.conf
   //
   //boot file for primary name server
   //
   //type domain source file or host
   //
   directory        "/var/named";
};

logging {
   category default {
      file "/var/log/in.named";
      severity info;
      print-category yes;
      print-severity yes;
   };
};

zone "biz.com" in {
   type master;
   allow-transfer{ 192.168.1/24 };
   allow-update{ 192.168.1/24; };
   check-names ignore;
   file "db.inside";
};

zone "1.168.192.in-addr.arpa" in {
   type master;
   allow-transfer{ 192.168.1/24; };
   allow-update{ 192.168.1/24; };
   check-names ignore;
   file "db.192.168.1";
};

zone "0.0.127.in-addr.arpa" in {
   type master;
   file "db.127.0.0";
};

zone "." in {
   type hint;
   file "named.ca";
};
```

Listing C: *A dump of /var/named/db.biz.com.log, which contains a log of dynamic updates*

```
;BIND LOG V8
[DYNAMIC_UPDATE] id 21 from [192.168.1.2].1056 at 974693484 (named pid 197):
zone: origin biz.com class IN serial 2820852217
update: {delete} fe9d89be-6d46-4491-99b9-50a9affc5c10._msdcs.biz.com. IN CNAME
update: {add} fe9d89be-6d46-4491-99b9-50a9affc5c10._msdcs.biz.com. 600 IN CNAME aphrodite.biz.com.

;BIND LOG V8
[INCR_SERIAL] from 2820852217 to 2820852218 Sun Nov 19 21:11:24 2000

;BIND LOG V8
[DYNAMIC_UPDATE] id 75 from [192.168.1.2].1167 at 974693820 (named pid 197):
zone: origin biz.com class IN serial 2820852218
update: {delete} fe9d89be-6d46-4491-99b9-50a9affc5c10._msdcs.biz.com. IN CNAME
update: {add} fe9d89be-6d46-4491-99b9-50a9affc5c10._msdcs.biz.com. 600 IN CNAME aphrodite.biz.com.

;BIND LOG V8
[INCR_SERIAL] from 2820852218 to 2820852219 Sun Nov 19 21:17:00 2000
```

On a Solaris host, which uses the /etc/named.conf file shown in this article, BIND creates the following four files in the /var/named directory:

```
db.192.168.1.ixfr
db.192.168.1.log
db.biz.com.ixfr
db.biz.com.log
```

Keep in mind that your own domain name and IP address appear in place of `biz.com` and `192.168.1`.

BIND logs zone updates to a file whose name ends with log. **Listing C** shows the contents of our db.biz.com.log, which BIND created. Notice how BIND increments the zone's serial number after each update.

Recent versions of BIND introduce incremental zone transfer (IXFR) protocol. IXFR allows slave servers to transfer only changed data instead of transferring the entire zone. BIND logs IXFR updates to a file whose name ends with ixfr. **Listing D** shows the contents of our db.biz.com.ixfr file.

Conclusion

Windows 2000 includes new Active Directory functionality that places additional demands upon Solaris' DNS. To support AD, your DNS server must accept SRV resource records. Additionally, it needs to allow underscores within SRV records. You also might want to enable dynamic updates so that AD can automatically create its SRV records for you. ✳

Listing D: *A dump of /var/named/db.biz.com.ixfr that logs the actions performed by incremental zone transfers*

```
;BIND LOG V8
[DYNAMIC_UPDATE] id 21 from [192.168.1.2].1056 at 974693484 (named pid 197):
zone:   origin biz.com class IN serial 2820852217
update: {delete} fe9d89be-6d46-4491-99b9-50a9affc5c10._msdcs.biz.com. 600 IN CNAME aphrodite.biz.com.
update: {add} fe9d89be-6d46-4491-99b9-50a9affc5c10._msdcs.biz.com. 600 IN CNAME aphrodite.biz.com.
;BIND LOG V8
update: {delete} biz.com. 86400 IN SOA apollo.biz.com. root.apollo.biz.com. ( 2820852217 10800 3600 604800 86400 )
update: {add} biz.com. 86400 IN SOA apollo.biz.com. root.apollo.biz.com. ( 2820852218 10800 3600 604800 86400 )
[END_DELTA]
;BIND LOG V8
[DYNAMIC_UPDATE] id 75 from [192.168.1.2].1167 at 974693820 (named pid 197):
zone:   origin biz.com class IN serial 2820852218
update: {delete} fe9d89be-6d46-4491-99b9-50a9affc5c10._msdcs.biz.com. 600 IN CNAME aphrodite.biz.com.
update: {add} fe9d89be-6d46-4491-99b9-50a9affc5c10._msdcs.biz.com. 600 IN CNAME aphrodite.biz.com.
;BIND LOG V8
update: {delete} biz.com. 86400 IN SOA apollo.biz.com. root.apollo.biz.com. ( 2820852218 10800 3600 604800 86400 )
update: {add} biz.com. 86400 IN SOA apollo.biz.com. root.apollo.biz.com. ( 2820852219 10800 3600 604800 86400 )
[END_DELTA]
```

Another database option: InterBase goes open source

by Clayton E. Crooks II

Application: InterBase 6
Operating systems: Solaris, Linux, Windows

In a statement directed to users of InterBase, Inprise (Borland) announced that they would make InterBase 6.0 available as an open-source database. As they promised, binary and source formats for the Linux, Windows and Solaris operating systems are now available for download, free of charge.

With version 6, Inprise has introduced several enhancements, including a higher level of SQL-92 compliance and a replication engine. The addition

of a commercial quality SQL database to the open-source movement is expected to please the thousands of developers who need the reliability, scalability and market-proven qualities of a product like InterBase, while maintaining the benefits of an open-source offering.

The open-source struggle

The decision to release the source code to version 6 of their popular InterBase database software under a custom version of the open-source Mozilla Public License 1.1 (MPL 1.1) was a great undertaking. In recent months, Inprise has been exposed to a plethora of setbacks, including a failed merger with Corel. In addition, the spin-off of InterBase has been delayed several times because of legal and financial issues, although the source code was released in July 2000 as promised.

The complete details of the InterBase custom version of the MPL 1.1, which is known as the InterBase Public License, are available at **www.InterBase.com/IPL.html**. For those unfamiliar with open-source licensing, the sidebar "Open-source licensing 101" provides a basic look at some standard licenses.

InterBase's history

InterBase has a long and winding history that has seen its ownership transferred on more than one occasion. It was originally founded in 1985, and operated without interference for almost 10 years, when it was first acquired. Ashton Tate held it for a short period of time until Inprise/Borland acquired it. InterBase's database once sold for as much as $10,000 per copy, but beginning with version 6, it's free.

Competition

Standards compliance is a crucial requirement for databases, particularly when they have to work with current code and data sources. Inprise's move to open their product's source to the world is significant, because it means InterBase will be the first fully standards-compliant, open-source database available. As a result, InterBase, which has been under development for more than 16 years, should be widely accepted by a variety of IT managers. It will be the first open-source database that can be embraced without having to make an impossible decision about the features you can live without. That

Open-source licensing 101

by Clayton E. Crooks II

Open-source licensing provides a wide variety of advantages to traditional licensing in a variety of ways. Unfortunately, the licensing itself is often difficult to understand. As a result, we've compiled a basic resource for established and successful approaches to open-source licensing, including the three most important licenses, presented from oldest to newest.

BSD licenses

Berkeley Standard Distribution (BSD) licenses are the oldest and least restrictive, as they give licensees the option to create private-derived works. The BSD license even allows vendors with traditional commercial software and unpublished source code to use BSD code. Any alterations of the code don't have to be given back to the community. BSD-licensed software has been used on projects such as Apache.

GNU General Public License

The GNU General Public License (GPL) is based on the Copyleft concept, which provides unlimited permission to copy and modify an original, and it also obligates the user to distribute the source code of all derivative works. The GPL-licensed projects can't be combined with work governed by different licenses. If you enhance a GPL work, your enhancements by default fall under the GPL terms. The GPL license has been the centerpiece of several applications, including the Linux kernel and the Samba SMB File Server.

Mozilla Public License

The Mozilla Public License (MPL or MozPL) was developed by Netscape Communications as part of their open-source release of the Communicator 5 Web browser. It's a totally different option, but you can see parts of the other two influencing it.

Changes made to an MPL source must be made freely available on the Web. Although MPL does allow for MPL sources that form a larger work to be licensed differently or remain unpublished.

Completely understanding open-source licensing would require volumes of information. As a result, if you have questions regarding licensing-related issues, we suggest you consult an appropriate legal advisor.

alone makes it a welcome addition to the open-source software movement.

Before the InterBase decision, several open-source databases were available, but PostgreSQL was the most established existing database. At this time, PostgreSQL has yet to reach SQL-92 entry-level compliance. It's continuing towards its compliance goal, but it currently lacks extensions such as declarative referential integrity and outer joins, not to mention a number of smaller issues with SQL compatibility.

Obviously, having everything in place for SQL-92 entry-level compliance is important, but a database also needs to perform at a high level of speed in order to be commercially useful. According to a variety of third-party tests, PostgreSQL also lags behind InterBase on most performance-related issues.

Conclusion

InterBase is a product that has been around since the 1980s and has sold for as much as $10,000 a seat. With the release of the open-source version, InterBase will have only service and support as a revenue stream, which in turn requires a substantial user base. If the large number of downloads from InterBase's Web site (**www.InterBase.com**) is any indication, this offering is widely accepted by Linux, Solaris and Windows developers. ✳

Image editing on Solaris with the GIMP

by Clayton E. Crooks II

Application: GIMP

The General Image Manipulation Program (or GNU Image Manipulation Program) is a free image-editing program that performs well on a variety of operating systems, including Solaris. The GIMP, as it's known, may be the most successful productivity application ever developed under an open source licensing arrangement. If you've previously worked with Adobe Photoshop or a similar type of graphics editor, you'll definitely be attracted to the features provided by the GIMP.

The beginning

Like many applications, the GIMP began rather unassumingly as a project for two students at the University of California at Berkeley, Spencer Kimball and Peter Mattis. Once the two had completed the course, they released the software under the GNU General Public License (GPL). At this time the software took off, as hundreds of programmers throughout the world began to improve the quality of the software by adding previously lacking features or fixing some of the early bugs. The GIMP continues to thrive several years later, and the users of the application have the dedicated work of volunteers from around the world to thank for the amazing capabilities of the program.

Although the program has been compiled on a wide range of operating systems, the software is open source, and as such, the source code can be obtained under the GPL for free. This can be a great advantage to end users, as it may allow them to use the application on a platform that hasn't previously been supported. At this time, the precompiled versions support a large number of operating systems, including Solaris.

You can download the GIMP for most of the available platforms at **www.gimp.org**. Many times, the software installation proceeds without interruption, but there are some things that can lead to problems with the installation. Because of these potential difficulties, you're advised to study the online manual available at **http://manual.gimp.org/** along with the installation help files at **www.gimp.org/install_help.html**.

The interface

The graphical user interface (GUI) doesn't stand out or detract from the software. In fact, it's very similar to Photoshop, which makes it very easy to learn if you've been a user of Adobe's software. As you can see in **Figure A** on the next page, when you launch the GIMP, a toolbox displays. The toolbox, which is the only item that's automatically displayed by the GIMP, contains all the

usual features you'd expect to find, including selection tools, various painting tools, and text tools. It looks and works in much the same way as Photoshop, although there are two additional buttons on the toolbox that provide the supplementary functions required by the GIMP for creating new images, opening images, and a variety of additional tasks.

The interface further molds into shape when you open or create an image. Once the image is open, you'll notice the lack of a menu bar, which keeps you guessing as to where the commands are located. However, simply right-clicking on an opened or freshly created image opens a pop-up window with a tremendous number of options. Between the toolbox and the pop-up window, you can locate all of the available tools and functions that the GIMP provides, which helps to make the user interface easy to learn and use.

Commercial-quality features

The GIMP provides a significant number of features that are comparable to commercial applications such as Photoshop. In fact, if you use Photoshop, you'll notice that many of the GIMP's features seem to have been based upon the Adobe application. Besides the toolbar and the GUI in general, many additional features have been borrowed from Photoshop. They include paint tools that offer comparable varieties of paintbrushes, an airbrush, and text tools; palettes that provide functions for dealing with channels and layers; and the selection tool that provides the ability to select images (or sections of images) by using the rectangle, ellipse or freehand tools.

A feature that can be surprisingly useful is the Pattern Fill, which enables you to fill a selection with a pattern instead of a color. To get to this option, right-click, select Dialogs | Patterns or File | Dialogs | Patterns to open the Pattern dialog box, where you can choose a fill pattern. There are some interesting patterns included with the GIMP, and if you can't find exactly what you need, you can create your own.

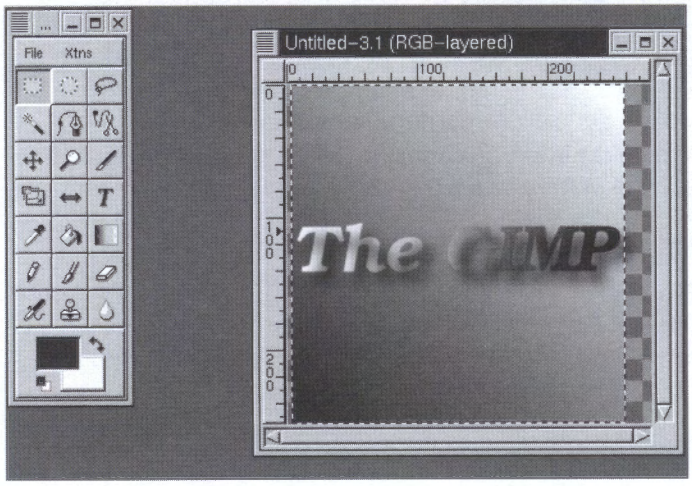

Figure A: *The GIMP interface is comparable to Photoshop.*

One of the most important features of a graphics application is the support it provides for the large variety of file formats, including PNG, TIFF, GIF, EPS, JPEG and even native Photoshop files. The GIMP is entirely supported on the Internet, so it isn't too surprising that it has extensive built-in support for creating Web graphics such as Animated GIFs. The construction of an animated GIF is particularly interesting, as the GIMP provides the ability to turn multiple layers into a single animated GIF.

The GIMP also supports plug-ins that are faithfully being developed by supporters all over the world. While many of the plug-ins would be useful only in very specific instances or not useful at all, a select few offer tremendous benefits that rival commercial plug-ins available for Photoshop. You can find a considerable list of plug-ins at **http://registry.gimp.org/**.

Script-fu

You've reviewed the majority of interesting features offered by the GIMP, but at this point in time you have yet to view its most powerful asset—Script-fu. Many people incorrectly refer to Script-fu as a type of macro, but in reality, it's a complete programming language that's infinitely more powerful than a simple macro system and is based on a programming language called Scheme.

Scheme is an interpreted programming language and works by querying functions to the GIMP database. Although there are an infinite number of possible uses for Script-fu, you'll most likely use it for automating things that are difficult or time-consuming to do, or tasks you wouldn't normally have the ability to accomplish. It's especially useful for automating tasks like batch file conversions.

Scheme is the default programming language, but it isn't the only scripting language available for the GIMP, as Perl and Tcl are also available scripting extensions. While most of the documentation and tutorials that are available center around the use of Scheme, you may want to use one of the others. This can be particularly advantageous if you've worked with either of the languages before. They are freely available, and you can download them at **http://registry.gimp.org**.

Many times, these types of add-ons are only useful if you devote a tremendous amount of time

and effort to them. And while it's true that if you have the time and patience to learn one of the languages supported by the GIMP you can develop some amazing scripts, it isn't a requirement to benefit from Script-fu, as it still offers some compelling advantages.

You can simply download and run one of the hundreds of prefabricated scripts that are freely available on the Web. These scripts can perform a myriad of tasks, and are obtainable because of the tremendous number of users who are willing to give back to the project. Installing a script is very easy, as you simply copy or move it to the .gimp/scripts directory, and it appears in your menus and can be used like any of the built-in options.

Script-fu is undoubtedly an amazing and well-thought-out idea, but there are some common mistakes that new GIMP users make when utilizing it. For instance, sometimes the scripts are dependent on certain actions taking place prior to their usage. This can be a simple dependency, such as the text tool needing to be selected, or it could be very involved. It's imperative that you read the directions included with the scripts, as most people assume that the script is simply bad if it's not working correctly. Another type of problem occurs when the script requires a certain item, such as a specific font, so be sure to read the Script dialog box before assuming it's not functioning.

A few problems

The GIMP is an excellent application, especially when considering the price you pay for it. And while the vast majority of the experiences with the software are positive, it does have a few problems. The first, and

About our contributors

Clayton E. Crooks II is a self-employed computer consultant living in Knoxville, Tenn. He's married with one child. His hobbies include game development, 3-D modeling and any athletic activity he can find time for.

Edgar Danielyan is currently self-employed. His qualifications include Cisco Certified Network Associate, diploma in company law from the British Institute of Legal Executives, and certified paralegal from the University of Southern Colorado. He's been working as a network administrator and manager of a top-level domain of Armenia. He's also worked for the United Nations, the ministry of defense, a national telco, a bank, and has been a partner in a law firm. He speaks four languages, likes good tea, and is a member of ACM, IEEE CS, USENIX, CIPS, ISOC and IPG, to name a few. He can be reached at **edd@danielyan.com**.

Arthur Haigh is a senior integration engineer at Destiny WebSolutions, a leading Internet consulting and digital business development firm focusing exclusively on e-financial services.

Don Kuenz works at Computing Resources Company (**gtcs.com/crc**). They provide programming, administration and hardware for Sun and PC platforms. You can reach Don at **kuenz@gtcs.com**.

Rob Thomas is a systems, network and security architect with the professional services division of a large telco. He can be reached at **robt@cymru.com**, or visit his home page at **www.enteract.com/~robt**.

Inside Solaris™
Tips & Techniques for users of Sun Solaris

Inside Solaris (ISSN 1081-3314) is published monthly by Element K Journals, a division of Element K Press, 500 Canal View Boulevard, Rochester, N.Y., 14624.

Customer Relations

U.S. toll free ..... (800) 223-8720
Outside of the U.S. ..... (716) 240-7301
Customer Relations fax ..... (716) 214-2386

For subscriptions, fulfillment questions, and requests for group subscriptions, address your letters to

Element K Journals Customer Relations
500 Canal View Boulevard
Rochester, NY 14623

Or contact Customer Relations via Internet email at journals@element-k.com.

Editorial

Editor ..... Garrett Suhm

Assistant Editor ..... Jill Suhm
Editorial Director ..... Michelle Rogers
Copy Editors ..... Rachel Krayer, Glenna Lechner
Contributing Editors ..... Clayton E. Crooks II, Edgar Danielyan, Arthur Haigh, Don Kuenz, Rob Thomas
Graphic Designer ..... Rachel J. King
Cover and Content Design ..... Melissa Ribaudo

You may address tips, special requests, and other correspondence to

The Editor, *Inside Solaris*
500 Canal View Boulevard
Rochester, NY 14623

Editorial Department fax ..... (716) 272-0064

Or contact us via Internet email at inside_solaris@elementkjournals.com.

Sorry, but due to the volume of mail we receive, we can't always promise a reply, although we do read every letter.

Element K Journals

General Manager ..... Kelly Baptiste
Manager of Customer Relations ..... Nicole Pate
Manager of Operations ..... Cristal Haygood
Manager of Graphic Design ..... Ian Caspersson
Manager of Product Marketing ..... Mike Mayfield
Senior Product Marketing Manager ..... Brian Cardona

Postmaster

Periodicals postage paid in Rochester, N.Y., and additional mailing offices.

Postmaster: Send address changes to

Inside Solaris
P.O. Box 92880
Rochester, NY 14692

Printed in the U.S.A.

Price

Domestic ..... $129/yr ($11.00 each)
Outside U.S. ..... $149/yr ($13.00 each)

Our Canadian GST# is: R140496720. CPM# is: 1446703.
QST# is: 1018491237.

Back Issues

To order a back issue from the last six months, call Customer Relations at (800) 223-8720. Back issues cost $11.00 each, $13.00 outside the U.S. You can pay with MasterCard, VISA, Discover or American Express.

Are you moving?

If you've moved recently or you're planning to move, you can guarantee uninterrupted service on your subscription by calling us at (800) 223-8720 and giving us your new address. Or you can fax us your label with the appropriate changes at (716) 214-2386. Our Customer Relations department is also available via email at journals@element-k.com.

Coming up...

- Internet protocols: ICMP
- Better encryption for Solaris 8

largest of these is the relatively large number of program bugs and errors.

As we previously mentioned, the software is open source. This benefits the GIMP in numerous ways, but there's a down side to this as well. First, there's not a single entity (individual or company) that's responsible for its development, and you often find bugs that would be repaired in commercial applications continue to remain for long periods of time, perhaps indefinitely. This isn't to suggest that the people maintaining the code don't care about these problems, it's just that many times a new feature will be added before a bug will even be looked at. The GIMP Web site has a complete list of known bugs, and some even have workarounds that aren't too difficult to use. The largest problems have already been taken care of by the hundreds of programmers updating the program.

A second area that can cause problems is the installation of the software. While the installations may work without a problem, many times a system isn't configured correctly or the end user isn't knowledgeable enough to complete the steps required for installation. Although the steps aren't tremendously difficult, they can offer a number of problems for individuals with little experience. This is especially true on the UNIX versions. You should pay particular attention to the installation instructions and follow them step by step.

One final area that can cause some distress for end users is directly related to the wide user base. Because the GIMP has been ported to a large number of platforms that are independent of one another, many of the distributions are lagging behind the others in features or quality. More often than not, the documentation doesn't reflect this information, which adds to the confusion. You may spend hours looking for a feature that's listed for one platform only to find out that it's unavailable in your particular distribution. You should check the current information for your operating system to see if any of the features are lagging behind or if there are updates available that will bring it up to date.

Conclusion

Overall, the GIMP is a top-notch graphics editor that provides a set of features that are nearly on par with those of Adobe Photoshop. In fact, the GIMP may actually be superior in some ways, as is the case with the Script-fu scripting tools. Script-fu combines a powerful language with the entire feature set of the GIMP to provide features that are unavailable even in the costly commercial graphics packages. You can download hundreds of freely available Script-fu scripts from the Internet. If you can't find one that will do exactly what you're looking for, you can create your own with a small investment in time. Script-fu frees you to do more of the work that you enjoy instead of spending hundreds of hours on mind-numbing tasks.

The tools that are included with the GIMP include all of the standard variety needed for image editing. But perhaps the most important feature of a graphics application is the support it provides for different file formats. The GIMP adequately covers this arena as well, through its ability to work with a large number of graphics files.

The GIMP showcases both the positives and negatives of the open source model of software distribution. There are several problems that can occur with installations, and distributions that are lagging behind the standard can cause problems in certain situations; however, they're generally not a factor. The quality of the freely available package is outstanding, and anyone looking for a UNIX-based graphics package should take the time to download and evaluate the GIMP. ✳