NAME

random, urandom − kernel random number source devices

DESCRIPTION

The character special files */dev/random* and */dev/urandom* (present since Linux 1.3.30) provide an interface to the kernel's random number generator. File */dev/random* has major device number 1 and minor device number 8. File */dev/urandom* has major device number 1 and minor device number 9.

The random number generator gathers environmental noise from device drivers and other sources into an entropy pool. The generator also keeps an estimate of the number of bits of noise in the entropy pool. From this entropy pool random numbers are created.

When read, the */dev/random* device will only return random bytes within the estimated number of bits of noise in the entropy pool. */dev/random* should be suitable for uses that need very high quality randomness such as one-time pad or key generation. When the entropy pool is empty, reads from */dev/random* will block until additional environmental noise is gathered.

A read from the */dev/urandom* device will not block waiting for more entropy. As a result, if there is not sufficient entropy in the entropy pool, the returned values are theoretically vulnerable to a cryptographic attack on the algorithms used by the driver. Knowledge of how to do this is not available in the current non-classified literature, but it is theoretically possible that such an attack may exist. If this is a concern in your application, use */dev/random* instead.

Usage

If you are unsure about whether you should use */dev/random* or */dev/urandom*, then probably you want to use the latter. As a general rule, */dev/urandom* should be used for everything except long-lived GPG/SSL/SSH keys.

If a seed file is saved across reboots as recommended above (all major Linux distributions have done this since 2000 at least), the output is cryptographically secure against attackers without local root access as soon as it is reloaded in the boot sequence, and perfectly adequate for network encryption session keys. Since reads from */dev/random* may block, users will usually want to open it in non-blocking mode (or perform a read with timeout), and provide some sort of user notification if the desired entropy is not immediately available.

The kernel random-number generator is designed to produce a small amount of high-quality seed material to seed a cryptographic pseudo-random number generator (CPRNG). It is designed for security, not speed, and is poorly suited to generating large amounts of random data. Users should be very economical in the amount of seed material that they read from */dev/urandom* (and */dev/random*); unnecessarily reading large quantities of data from this device will have a negative impact on other users of the device.

The amount of seed material required to generate a cryptographic key equals the effective key size of the key. For example, a 3072-bit RSA or Diffie-Hellman private key has an effective key size of 128 bits (it requires about 2^128 operations to break) so a key generator only needs 128 bits (16 bytes) of seed material from */dev/random*.

While some safety margin above that minimum is reasonable, as a guard against flaws in the CPRNG algorithm, no cryptographic primitive available today can hope to promise more than 256 bits of security, so if any program reads more than 256 bits (32 bytes) from the kernel random pool per invocation, or per reasonable re-seed interval (not less than one minute), that should be taken as a sign that its cryptography is *not* skilfully implemented.

Configuration

If your system does not have */dev/random* and */dev/urandom* created already, they can be created with the following commands:

```
mknod −m 644 /dev/random c 1 8
mknod −m 644 /dev/urandom c 1 9
```

```
chown root:root /dev/random /dev/urandom
```

When a Linux system starts up without much operator interaction, the entropy pool may be in a fairly predictable state. This reduces the actual amount of noise in the entropy pool below the estimate. In order to counteract this effect, it helps to carry entropy pool information across shut-downs and start-ups. To do this, add the following lines to an appropriate script which is run during the Linux system start-up sequence:

```
echo "Initializing random number generator..."
random_seed=/var/run/random-seed
# Carry a random seed from start-up to start-up
# Load and then save the whole entropy pool
if [ −f $random_seed ]; then
    cat $random_seed >/dev/urandom
else
    touch $random_seed
fi
chmod 600 $random_seed
poolfile=/proc/sys/kernel/random/poolsize
[ −r $poolfile ] && bytes=`cat $poolfile` || bytes=512
dd if=/dev/urandom of=$random_seed count=1 bs=$bytes
```

Also, add the following lines in an appropriate script which is run during the Linux system shutdown:

```
# Carry a random seed from shut-down to start-up
# Save the whole entropy pool
echo "Saving random seed..."
random_seed=/var/run/random-seed
touch $random_seed
chmod 600 $random_seed
poolfile=/proc/sys/kernel/random/poolsize
[ −r $poolfile ] && bytes=`cat $poolfile` || bytes=512
dd if=/dev/urandom of=$random_seed count=1 bs=$bytes
```

/proc Interface

The files in the directory */proc/sys/kernel/random* (present since 2.3.16) provide an additional interface to the */dev/random* device.

The read-only file *entropy_avail* gives the available entropy. Normally, this will be 4096 (bits), a full entropy pool.

The file *poolsize* gives the size of the entropy pool. The semantics of this file vary across kernel versions:

- Linux 2.4: This file gives the size of the entropy pool in *bytes*. Normally, this file will have the value 512, but it is writable, and can be changed to any value for which an algorithm is available. The choices are 32, 64, 128, 256, 512, 1024, or 2048.
- Linux 2.6: This file is read-only, and gives the size of the entropy pool in *bits*. It contains the value 4096.

The file *read_wakeup_threshold* contains the number of bits of entropy required for waking up processes that sleep waiting for entropy from */dev/random*. The default is 64. The file *write_wakeup_threshold* contains the number of bits of entropy below which we wake up processes that do a **select**(2) or **poll**(2) for write access to */dev/random*. These values can be changed by writing to the files.

The read-only files *uuid* and *boot_id* contain random strings like 6fd5a44b-35f4-4ad4-a9b9-6b9be13e1fe9. The former is generated afresh for each read, the latter was generated once.

FILES

/dev/random
/dev/urandom

SEE ALSO

mknod (1)
RFC 1750, "Randomness Recommendations for Security"

COLOPHON

This page is part of release 3.22 of the Linux *man-pages* project. A description of the project, and information about reporting bugs, can be found at http://www.kernel.org/doc/man-pages/.