NAME

ttymon – port monitor for terminal ports

SYNOPSIS

/usr/lib/saf/ttymon
/usr/lib/saf/ttymon **–g** [**–d** *device*] [**–h**] [**–t** *timeout*] [**–l** *ttylabel*] [**–p** *prompt*] [**–m** *modules*] [**–T** *termtype*]

DESCRIPTION

ttymon is a STREAMS-based TTY port monitor. Its function is to monitor ports, to set terminal modes, baud rates, and line disciplines for the ports, and to connect users or applications to services associated with the ports. Normally, **ttymon** is configured to run under the Service Access Controller, **sac**(1M), as part of the Service Access Facility (SAF). It is configured using the **sacadm**(1M) command. Each instance of **ttymon** can monitor multiple ports. The ports monitored by an instance of **ttymon** are specified in the port monitor's administrative file. The administrative file is configured using the **pmadm**(1M) and **ttyadm**(1M) commands. When an instance of **ttymon** is invoked by the **sac** command, it starts to monitor its ports. For each port, **ttymon** first initializes the line disciplines, if they are specified, and the speed and terminal settings. For ports with entries in **/etc/logindevperm**, device owner, group and permissions are set. (See **logindevperm**(4).) The values used for initialization are taken from the appropriate entry in the TTY settings file. This file is maintained by the **sttydefs**(1M) command. Default line disciplines on ports are usually set up by the **autopush**(1M) command of the Autopush Facility.

ttymon then writes the prompt and waits for user input. If the user indicates that the speed is inappropriate by pressing the BREAK key, **ttymon** tries the next speed and writes the prompt again. When valid input is received, **ttymon** interprets the per-service configuration file for the port, if one exists, creates a **utmp** entry if required (see **utmp**(4)), establishes the service environment, and then invokes the service associated with the port. Valid input consists of a string of at least one non-newline character, terminated by a carriage return. After the service terminates, **ttymon** cleans up the **utmp** entry, if one exists, and returns the port to its initial state.

If *autobaud* is enabled for a port, **ttymon** will try to determine the baud rate on the port automatically. Users must enter a carriage return before **ttymon** can recognize the baud rate and print the prompt. Currently, the baud rates that can be determined by *autobaud* are **110**, **1200**, **2400**, **4800**, and **9600**.

If a port is configured as a bidirectional port, **ttymon** will allow users to connect to a service, and, if the port is free, will allow **uucico**(1M), **cu**(1C), or **ct**(1C) to use it for dialing out. If a port is bidirectional, **ttymon** will wait to read a character before it prints a prompt.

If the *connect-on-carrier* flag is set for a port, **ttymon** will immediately invoke the port's associated service when a connection request is received. The prompt message will not be sent.

If a port is disabled, **ttymon** will not start any service on that port. If a disabled message is specified, **ttymon** will send out the disabled message when a connection request is received. If **ttymon** is disabled, all ports under that instance of **ttymon** will also be disabled.

SERVICE INVOCATION

The service **ttymon** invokes for a port is specified in the **ttymon** administrative file. **ttymon** will scan the character string giving the service to be invoked for this port, looking for a **%d** or a **%%** two-character sequence. If **%d** is found, **ttymon** will modify the service command to be executed by replacing those two characters by the full path name of this port (the device name). If **%%** is found, they will be replaced by a single **%**.

When the service is invoked, file descriptor **0**, **1**, and **2** are opened to the port device for reading and writing. The service is invoked with the user ID, group ID and current home directory set to that of the user name under which the service was registered with **ttymon**. Two environment variables, **HOME** and **TTYPROMPT,** are added to the service's environment by **ttymon**. **HOME** is set to the home directory of the user name under which the service is invoked. **TTYPROMPT** is set to the prompt string configured for the service on the port. This is provided so that a service invoked by **ttymon** has a means of determining if a prompt was actually issued by **ttymon** and, if so, what that prompt actually was.

See **ttyadm**(1M) for options that can be set for ports monitored by **ttymon** under the Service Access Controller.

OPTIONS

–g A special invocation of **ttymon** is provided with the **–g** option. This form of the command should only be called by applications that need to set the correct baud rate and terminal settings on a port and then connect to **login** service, but that cannot be pre-configured under the SAC. The following combinations of options can be used with **–g**:

–d *device* *device* is the full path name of the port to which **ttymon** is to attach. If this option is not specified, file descriptor **0** must be set up by the invoking process to a TTY port.

–h If the -h flag is not set, **ttymon** will force a hangup on the line by setting the speed to zero before setting the speed to the default or specified speed.

–l *ttylabel* *ttylabel* is a link to a speed and TTY definition in the **ttydefs** file. This definition tells **ttymon** at what speed to run initially, what the initial TTY settings are, and what speed to try next if the user indicates that the speed is inappropriate by pressing the BREAK key. The default speed is 9600 baud.

–m *modules* When initializing the port, **ttymon** will pop all modules on the port, and then push *modules* in the order specified. *modules* is a comma-separated list of pushable modules. Default modules on the ports are usually set up by the Autopush Facility.

–p *prompt* Allows the user to specify a prompt string. The default prompt is **Login:**

.

–t *timeout* Specifies that **ttymon** should exit if no one types anything in *timeout* seconds after the prompt is sent.

–T *termtype* Sets the **TERM** environment variable to *termtype*.

ENVIRONMENT

If any of the **LC_**∗ variables (**LC_CTYPE, LC_MESSAGES, LC_TIME, LC_COLLATE, LC_NUMERIC,** and **LC_MONETARY**) (see **environ**(5)) are not set in the environment, the operational behavior of **ttymon** for each corresponding locale category is determined by the value of the **LANG** environment variable. If **LC_ALL** is set, its contents are used to override both the **LANG** and the other **LC_**∗ variables. If none of the above variables is set in the environment, the "C" (U.S. style) locale determines how **ttymon** behaves.

LC_CTYPE
Determines how **ttymon** handles characters. When **LC_CTYPE** is set to a valid value, **ttymon** can display and handle text and filenames containing valid characters for that locale. **ttymon** can display and handle Extended Unix Code (EUC) characters where any individual character can be 1, 2, or 3 bytes wide. **ttymon** can also handle EUC characters of 1, 2, or more column widths. In the "C" locale, only characters from ISO 8859-1 are valid.

FILES

/etc/logindevperm

SEE ALSO

ct(1C), **cu**(1C), **autopush**(1M), **pmadm**(1M), **sac**(1M), **sacadm**(1M), **sttydefs**(1M), **ttyadm**(1M), **uucico**(1M), **logindevperm**(4), **utmp**(4), **environ**(5)

System Administration Guide, Volume II

NOTES

If a port is monitored by more than one **ttymon**, it is possible for the **ttymon**s to send out prompt messages in such a way that they compete for input.