```
**********************************************************
  219319 Tue May  6 16:13:01 2014
new/usr/src/uts/common/inet/ip/sadb.c
4851 IPsec should use NANOSEC & friends from time.h
**********************************************************
______unchanged_portion_omitted_

4214 /*
4215  * Figure out when to reschedule the ager.
4216  */
4217 timeout_id_t
4218 sadb_retimeout(hrtime_t begin, queue_t *pfkey_q, void (*ager)(void *),
4219     void *agerarg, uint_t *intp, uint_t intmax, short mid)
4220 {
4221         hrtime_t end = gethrtime();
4222         uint_t interval = *intp;        /* "interval" is in ms. */
4222         uint_t interval = *intp;

4224         /*
4225          * See how long this took.  If it took too long, increase the
4226          * aging interval.
4227          */
4228         if ((end - begin) > MSEC2NSEC(interval)) {
4228         if ((end - begin) > (hrtime_t)interval * (hrtime_t)1000000) {
4229                 if (interval >= intmax) {
4230                         /* XXX Rate limit this?  Or recommend flush? */
4231                         (void) strlog(mid, 0, 0, SL_ERROR | SL_WARN,
4232                             "Too many SA's to age out in %d msec.\n",
4233                             intmax);
4234                 } else {
4235                         /* Double by shifting by one bit. */
4236                         interval <<= 1;
4237                         interval = min(interval, intmax);
4238                 }
4239         } else if ((end - begin) <= (MSEC2NSEC(interval) / 2) &&
4239         } else if ((end - begin) <= (hrtime_t)interval * (hrtime_t)500000 &&
4240             interval > SADB_AGE_INTERVAL_DEFAULT) {
4241                 /*
4242                  * If I took less than half of the interval, then I should
4243                  * ratchet the interval back down.  Never automatically
4244                  * shift below the default aging interval.
4245                  *
4246                  * NOTE:This even overrides manual setting of the age
4247                  *      interval using NDD to lower the setting past the
4248                  *      default.  In other words, if you set the interval
4249                  *      lower than the default, and your SADB gets too big,
4250                  *      the interval will only self-lower back to the default.
4251                  */
4252                 /* Halve by shifting one bit. */
4253                 interval >>= 1;
4254                 interval = max(interval, SADB_AGE_INTERVAL_DEFAULT);
4255         }
4256         *intp = interval;
4257         return (qtimeout(pfkey_q, ager, agerarg,
4258             drv_usectohz(interval * (MICROSEC / MILLISEC))));
4258             drv_usectohz(interval * 1000)));
4259 }
______unchanged_portion_omitted_
```

```
**********************************************************
  190741 Tue May  6 16:13:01 2014
new/usr/src/uts/common/inet/ip/spd.c
4851 IPsec should use NANOSEC & friends from time.h
**********************************************************
______unchanged_portion_omitted_

1052 /*
1053  * Rate-limiting front-end to strlog() for AH and ESP.  Uses the ndd variables
1054  * in /dev/ip and the same rate-limiting clock so that there's a single
1055  * knob to turn to throttle the rate of messages.
1056  */
1057 void
1058 ipsec_rl_strlog(netstack_t *ns, short mid, short sid, char level, ushort_t sl,
1059     char *fmt, ...)
1060 {
1061         va_list adx;
1062         hrtime_t current = gethrtime();
1063         ip_stack_t      *ipst = ns->netstack_ip;
1064         ipsec_stack_t   *ipss = ns->netstack_ipsec;

1066         sl |= SL_CONSOLE;
1067         /*
1068          * Throttle logging to stop syslog from being swamped. If variable
1069          * 'ipsec_policy_log_interval' is zero, don't log any messages at
1070          * all, otherwise log only one message every 'ipsec_policy_log_interval'
1071          * msec. Convert interval (in msec) to hrtime (in nsec).
1072          */

1074         if (ipst->ips_ipsec_policy_log_interval) {
1075                 if (ipss->ipsec_policy_failure_last +
1076                     MSEC2NSEC(ipst->ips_ipsec_policy_log_interval) <= current) {
1076                     ((hrtime_t)ipst->ips_ipsec_policy_log_interval *
1077                     (hrtime_t)1000000) <= current) {
1077                         va_start(adx, fmt);
1078                         (void) vstrlog(mid, sid, level, sl, fmt, adx);
1079                         va_end(adx);
1080                         ipss->ipsec_policy_failure_last = current;
1081                 }
1082         }
1083 }
______unchanged_portion_omitted_
```