                Algorithms for Synchronizing Network Clocks

Status of this Memo

   This RFC discussed clock synchronization algorithms for the
   ARPA-Internet community, and requests discussion and suggestions for
   improvements. Distribution of this memo is unlimited.

Table of Contents

List of Tables

1. Introduction

    The recent interest within the Internet community in determining
    accurate time from a set of mutually suspicious network clocks has
    been prompted by several occasions in which gross errors were found
    in usually reliable, highly accurate clock servers after seasonal
    thunderstorms which disrupted their primary power supply. To these
    sources of error should be added those due to malfunctioning
    hardware, defective software and operator mistakes, as well as random
    errors in the mechanism used to set and/or synchronize the clocks via
    Internet paths. The results of these errors can range from simple
    disorientation to major disruption, depending upon the operating
    system, when files or messages are incorrectly timestamped or the
    order of critical network transactions is altered.

    This report suggests a stochastic model based on the principles of
    maximum-likelihood estimation, together with algorithms for computing
    a good estimator from a number of time-offset samples measured
    between one or more clocks connected via network links. The model
    provides a rational method for detecting and resolving errors due to
    faulty clocks or excessively noisy links. Included in this report
    are descriptions of certain experiments conducted with Internet hosts
    and ARPANET paths which give an indication of the effectiveness of
    the algorithms.

    Several mechanisms have been specified in the Internet protocol suite
    to record and transmit the time at which an event takes place,
    including the ICMP Timestamp message [6], Time Protocol [7], Daytime
    protocol [8] and IP Timestamp option [9]. A new Network Time
    Protocol [12] has been proposed as well. Additional information on
    network time synchronization can be found in the References at the
    end of this document. Synchronization protocols are described in [3]
    and [12] and synchronization algorithms in [2], [5] and [10].
    Experimental results on measured roundtrip delays and clock offsets
    in the Internet are discussed in [4] and [11]. A comprehensive
    mathematical treatment of clock synchronization can be found in [1].

    In [10] the problem of synchronizing a set of mutually suspicious
    clocks is discussed and algorithms offered which maximize in some
    sense the expectation that a correct set of "good" clocks can be
    extracted from the population including also "bad" ones. The
    technique is based upon overlapping, discrete confidence intervals
    which are assigned a-priori. The model assumes the reasonable
    presumption that "bad" clocks display errors far outside these
    confidence intervals, so can be easily identified and discarded from
    the voting process.

   As apparent from the data summarized in Appendix A, host clocks in a
   real network commonly indicate various degrees of dispersion with
   respect to each other and to a standard-time reference such as a
   radio clock. The sources of dispersion include random errors due to
   observational phenomena and the synchronization mechanism itself, if
   used, as well as systematic errors due to hardware or software
   failure, poor radio reception conditions or operator mistakes. In
   general, it is not possible to accurately quantify whether the
   dispersion of any particular clock qualifies the clock as "good" or
   "bad," except on a statistical basis. Thus, from a practical
   standpoint, a statistical-estimation approach to the problem is
   preferred over a discrete-decision one.

   A basic assumption in this report is that the majority of "good"
   clocks display errors clustered around a zero offset relative to
   standard time, as determined for example from a radio clock, while
   the remaining "bad" clocks display errors distributed randomly over
   the observing interval. The problem is to select the good clocks
   from the bad and to estimate the correction to apply to the local
   clock in order to display the most accurate time. The algorithms
   described in this report attempt to do this using maximum-likelihood
   techniques, which are theory.

   It should be noted that the algorithms discussed in [10] and in this
   report are are basically filtering and smoothing algorithms and can
   result in errors, sometimes gross ones, if the sample distribution
   departs far from a-priori assumptions. Thus, a significant issue in
   the design of these algorithms is robustness in the face of skewed
   sample data sets. The approach in [10] uses theorem-proving to
   justify the robustness of the discrete algorithms presented;
   however, the statistical models in this report are not suited for
   that. The approach taken in this report is based on detailed
   observation and experiments, a summary of which is included as an
   appendix. While this gives an excellent qualitative foundation upon
   which to judge robustness, additional quantitative confidence should
   be developed through the use of statistical mechanics.

2. Majority-Subset Algorithms

   A stochastic model appropriate to a system of mutually suspicious
   clocks can be constructed as follows. An experiment consists of one
   or more measurements of time differences or offsets between several
   clocks in the network. Usually, but not necessarily, one of the
   clocks is the local clock at the observer and observations are
   conducted with each of several other clocks in the network. The fact
   that some clocks are presumed more accurate or trusted more highly
   than others can be expressed by weighting the measurements

accordingly. The result is a set of statistics, including means and
variances, from which the observer is able to estimate the best time
at which to set the local clock.

A maximum-likelihood estimator is a statistic that maximizes the
probability that a particular outcome of an experiment is due to a
presumed set of assumptions on the constraints of the experiment.
For example, if it is assumed that at least k of n observations
include only samples from a single distribution, then a
maximum-likelihood estimator for the mean of that distribution might
be computed as follows: Determine the variance for every k-sample
subset of the n observations. Then select the subset with smallest
variance and use its mean as the estimator for the distribution mean.

For instance, let n be the number of clocks and k be the next largest
integer in n/2, that is, the minimum majority. A majority subset is
a subset consisting of k of the n offset measurements. Each of these
subsets can be represented by a selection of k out of n
possibilities, with the total number of subsets equal to C(n,k). The
number of majority subsets is tallied for n from 2 to 20 in Table 1.

(n,k)	C(n,k)	(n,k)	C(n,k)
(2,2)	1	(11,6)	462
(3,2)	3	(12,7)	792
(4,3)	4	(13,7)	1716
(5,3)	10	(14,8)	3003
(6,4)	15	(15,8)	6435
(7,4)	35	(16,9)	11440
(8,5)	56	(17,9)	24310
(9,5)	126	(18,10)	43758
(10,6)	210	(19,10)	92378
		(20,11)	167960

Table 1. C(n,k) for n from 2 to 20

Obviously, the number of computations required becomes awkward as n
increases beyond about 10. Representative majority subsets for n =
3,4,5 are shown in Table 2.

```
        C(3,2)          C(4,3)          C(5,3)
        ------          ------          ------
        1,2             1,2,3           1,2,3
        1,3             1,2,4           1,2,4
        2,3             1,3,4           1,2,5
                        2,3,4           1,3,4
                                        1,3,5
                                        1,4,5
                                        2,3,4
                                        2,3,5
                                        2,4,5
                                        3,4,5
```

            Table 2. Majority Subsets for n = 3,4,5

   Choosing n = 5, for example, requires calculation of the mean and
   variance for ten subsets indexed as shown in Table 2.

   A maximum-likelihood algorithm with provision for multiple samples
   and weights might operate as follows: Let n be the number of clocks
   and w(1),w(2),...,w(n) a set of integer weights, with w(i) the weight
   associated with the ith clock. For the ith clock three accumulators
   W(i), X(i) and Y(i) are provided, each initialized to zero. The ith
   clock is polled some number of times, with each reply x causing n(i)
   to be added to W(i), as well as the weighted sample offset n(i)*x
   added to X(i) and its square (n(i)*x)2 added to Y(i). Polling is
   continued for each of the n clocks in turn.

   Next, using a majority-subset table such as shown in Table 2,
   calculate the total weight W = sum(W(i)) and weighted sums X =
   sum(X(i)) and Y = sum(Y(i)*Y(i)) for each i in the jth majority
   subset (row). From W, X and Y calculate the mean m(j) and variance
   s(j):

            m(j) = X/W   and   s(j) = Y/W - m(j)*m(j) .

   When this is complete for all rows, select the row j with the
   smallest s(j) and return the associated mean m(j) as the
   maximum-likelihood estimate of the local-clock offset.

3. Clustering Algorithms

   Another method for developing a maximum-likelihood estimator is
   through the use of clustering algorithms. These algorithms operate
   to associate points in a sample set with clusters on the basis of
   stochastic properties and are most useful when large numbers of
   samples are available. One such algorithm operates on a sample set
   to selectively discard points presumed outside the cluster as
   follows:

      1. Start with a sample set of n observations {x(1),x(2),...,x(n)

      2. Compute the mean of the n observations in the sample set.
          Discard the single sample x(i) with value furthest from the
          mean, leaving n-1 observations in the set.

      3. Continue with step 2 until only a single observation is left,
          at which point declare its value the maximum-likelihood
          estimator.

   This algorithm will usually (but not necessarily) converge to the
   desired result if the majority of observations are the result of
   "good" clocks, which by hypothesis are clustered about zero offset
   relative to the reference clock, with the remainder scattered
   randomly over the observation interval.

   The following Table 3 summarizes the results of this algorithm
   applied to the UDP data shown in Appendix A, which represents the
   measured clock offsets of 163 host clocks in the Internet system.
   These data were assembled using the UDP Time protocol [7], in which
   time is represented to a precision of one second. Each line of the
   table represents the result of step 2 above along with the size of
   the sample set and its (unweighted) mean and variance. The "Discard"
   column shows the value of the observation discarded at that step.

```
              Size    Mean    Var     Discard
              ------------------------------
              163     -210    9.1E+6  -38486
              162     26      172289  3728
              161     3       87727   3658
              160     -20     4280    -566
              150     -17     1272    88
              100     -18     247     -44
              50      -4      35      8
              20      -1      0       -2
              19      -1      0       -2
              18      -1      0       -2
              17      -1      0       1
              16      -1      0       -1
              15      -1      0       -1
              14      -1      0       -1
              13      0       0       0
              1       0       0       0
```

        Table 3. Clustering Algorithm using UDP Time Protocol Data

In Table 3 only a few of the 163 steps are shown, including those
near the beginning which illustrate a rapid convergence as the
relatively few outliers are discarded. The large outlier discarded
in the first step is almost certainly due to equipment or operator
failure. The two outliers close to one hour discarded in the next two
steps are probably simple operator mistakes like entering summer time
instead of standard time. By the time only 50 samples are left, the
error has shrunk to about 4 sec and the largest outlier is within 12
sec of the estimate. By the time only 20 samples are left, the error
has shrunk to about a second and the variance has vanished for
practical purposes.

The following Table 4 summarizes the results of the clustering
algorithm applied to the ICMP data shown in Appendix A, which
represents the measured clock offsets of 504 host clocks in the
Internet system. These data were assembled using ICMP Timestamp
messages [6], in which time is represented to a precision of one
millisecond. The columns in Table 4 should be interpreted in the
same way as in Table 3, except that the data in Table 4 are in
milliseconds, while the data in Table 3 are in seconds.

Size	Mean	Var	Discard
504	-3.0E+6	3.2E+14	8.6E+7
500	-3.3E+6	2.9E+14	8.6E+7
450	-1.6E+6	3.0E+13	-2.5E+7
400	29450	2.2E+11	3.6E+6
350	-3291	4.1E+9	-185934
300	3611	1.6E+9	-95445
250	2967	6.8E+8	66743
200	4047	2.3E+8	39288
150	1717	8.6E+7	21346
100	803	1.9E+7	10518
80	1123	8.4E+6	-4863
60	1119	3.1E+6	4677
50	502	1.5E+6	-2222
40	432	728856	2152
30	84	204651	-987
20	30	12810	338
15	28	2446	122
10	7	454	49
8	-2	196	24
6	-9	23	0
4	-10	5	-13
2	-8	0	-8

Table 4. Clustering Algorithm using ICMP Timestamp Data

   As in Table 3 above, only some of the 504 steps are shown in Table 4.
   The distinguishing feature of the data in Table 4 is that the raw
   data are much more noisy - only some 30 host clocks are closer than
   one second from the reference clock, while half were further than one
   minute and over 100 further than one hour from it. The fact that the
   algorithm converged to within 8 msec of the reference time under
   these conditions should be considered fairly remarkable in view of
   the probability that many of the outliers discarded are almost
   certainly due to defective protocol implementations. Additional
   information on these experiments is presented in Appendix A.

4. Application to Time-Synchronization Data

   A variation of the above algorithms can be used to improve the offset
   estimates from a single clock by discarding noise samples produced by
   occasional retransmissions in the network, for example. A set of n
   independent samples is obtained by polling the clock. Then, a
   majority-subset table is used to compute the $m(j)$ and $s(j)$ statistics
   and the maximum-likelihood estimate determined as above. For this
   purpose the majority-subset table could include larger subsets as
   well. In this manner the maximum-likelihood estimates from each of
   several clocks can be determined and used in the algorithm above.

   In order to test the effectiveness of this algorithm, a set of
   experiments was performed using two WIDEBAND/EISN gateways equipped
   with WWVB radio clocks and connected to the ARPANET. These
   experiments were designed to determine the limits of accuracy when
   comparing these clocks via ARPANET paths. One of the gateways
   (ISI-MCON-GW) is located at the Information Sciences Institute near
   Los Angeles, while the other (LL-GW) is located at Lincoln
   Laboratories near Boston. Both gateways consist of PDP11/44
   computers running the EPOS operating system and clock-interface
   boards with oscillators phase-locked to the WWVB clock.

   The clock indications of the WIDEBAND/EISN gateways were compared
   with the DCNet WWVB reference clock using ICMP Timestamp messages,
   which record the individual timestamps with a precision of a
   millisecond. However, the path over the ARPANET between these
   gateways and the measurement host can introduce occasional
   measurement errors as large as several seconds. In principle the
   effect of these errors can be minimized by using a large sample
   population; however, use of the above algorithms allows higher
   accuracies to be obtained with far fewer samples.

   Measurements were made separately with each of the two gateways by
   sending an ICMP Timestamp Request message from the ARPANET address of
   DCN1 to the ARPANET address of the gateway and computing the
   round-trip delay and clock offset from the ICMP Timestamp Reply
   message. This process was continued for 1000 message exchanges,
   which took from seven minutes to several hours, depending on the
   sample interval selected.

   The tables below summarize the results of both the majority-subset
   and clustering algorithms applied to the data from three experiments,
   one with ISI-MCON-GW and two with LL-GW. The ISI-MCON-GW and LL-GW
   (a) experiments were designed to determine the limits of accuracy
   when using a continuous sequence of request/reply volleys, which

resulted in over two samples per second. The remaining LL-GW (b)
experiment was designed to determine the limits of accuracy using a
much lower rate of about one sample every ten seconds.

For each of the three experiments two tables are shown, one using the
majority-subset algorithm and the other the clustering algorithm. The
two rows of the majority-subset tables show the statistics derived
both from the raw data and from the filtered data processed by a
C(5,3) majority-subset algorithm. In all cases the extrema and
variance are dramatically less for the filtered data than the raw
data, lending credence to the conjecture that the mean statistic for
the filtered data is probably a good maximum-likelihood estimator of
the true offset.

	Mean	Var	Max	Min
Raw data	637	2.1E+7	32751	-1096
C(5,3)	-15	389	53	-103

Table 5. ISI-MCON-GW Majority-Subset Algorithm

Size	Mean	Var	Discard
1000	637	2.1E+7	32751
990	313	1.0E+7	32732
981	15	1.0E+6	32649
980	-18	2713	-1096
970	-15	521	-122
960	-15	433	-97
940	-15	332	-75
900	-15	239	26
800	-15	141	12
700	-16	87	5
600	-17	54	-31
500	-16	33	-5
400	-18	18	-9
300	-19	7	-12
200	-19	2	-21
100	-18	0	-19
1	-17	0	-17

Table 6. ISI-MCON-GW Clustering Algorithm

	Mean	Dev	Max	Min
Raw data	566	1.8E+7	32750	-143
C(5,3)	-23	81	14	-69

Table 7. LL-GW (a) Majority-Subset Algorithm

Size	Mean	Var	Discard
1000	566	1.8E+7	32750
990	242	8.5E+6	32726
983	10	1.0E+6	32722
982	-23	231	-143
980	-23	205	-109
970	-22	162	29
960	-23	128	13
940	-23	105	-51
900	-24	89	1
800	-25	49	-9
700	-26	31	-36
600	-26	21	-34
500	-27	14	-20
400	-29	7	-23
300	-30	3	-33
200	-29	1	-27
100	-29	0	-28
1	-29	0	-29

Table 8. LL-GW (a) Clustering Algorithm

	Mean	Dev	Max	Min
Raw data	378	2.1E+7	32760	-32758
C(5,3)	-21	1681	329	-212

Table 9. LL-GW (b) Majority-Subset Algorithm

```
              Size    Mean    Var      Discard
              -------------------------------
              1000    377     2.1E+7   -32758
              990     315     1.0E+7   32741
              981     18      1.1E+6   32704
              980     -16     16119    -1392
              970     -17     5382     554
              960     -21     3338     311
              940     -24     2012     168
              900     -22     1027     -137
              800     -23     430      -72
              700     -23     255      -55
              600     -22     167      -45
              500     -22     109      -40
              400     -21     66       -6
              300     -18     35       -29
              200     -17     15       -23
              100     -19     3        -15
              50      -21     0        -19
              20      -21     0        -21
              10      -20     0        -20
              1       -20     0        -20
```

             Table 10. LL-GW (b) Clustering Algorithm

   The rows of the clustering tables show the result of selected steps
   in the algorithm as it discards samples furthest from the mean. The
   first twenty steps or so discard samples with gross errors over 30
   seconds. These samples turned out to be due to a defect in the
   timestamping procedure implemented in the WIDEBAND/EISN gateway code
   which caused gross errors in about two percent of the ICMP Timestamp
   Reply messages. These samples were left in the raw data as received
   in order to determine how the algorithms would behave in such extreme
   cases. As apparent from the tables, both the majority-subset and
   clustering algorithms effectively coped with the situation.

   In the statement of the clustering algorithm the terminating
   condition was specified as when only a single sample is left in the
   sample set. However, it is not necessary to proceed that far. For
   instance, it is known from the design of the experiment and the
   reference clocks that accuracies better than about ten milliseconds
   are probably unrealistic. A rough idea of the accuracy of the mean
   is evident from the deviation, computed as the square root of the
   variance. Thus, attempts to continue the algorithm beyond the point
   where the variance drops below 100 or so are probably misguided.
   This occurs when between 500 and 900 samples remain in the sample

set, depending upon the particular experiment. Note that in any case
between 300 and 700 samples fall within ten milliseconds of the final
estimate, depending on experiment.

Comparing the majority-subset and clustering algorithms on the basis
of variance reveals the interesting observation that the result of
the C(5,3) majority-subset algorithm is equivalent to the clustering
algorithm when between roughly 900 and 950 samples remain in the
sample set. This together with the moderately high variance in the
ISI-MCON-GW and LL-GW (b) cases suggests a C(6,4) or even C(7,4)
algorithm might yield greater accuracies.

5. Summary and Conclusions

The principles of maximum-likelihood estimation are well known and
widely applied in communication electronics. In this note two
algorithms using these principles are proposed, one based on
majority-subset techniques appropriate for cases involving small
numbers of samples and the other based on clustering techniques
appropriate for cases involving large numbers of samples.

The algorithms were tested on raw data collected with Internet hosts
and gateways over ARPANET paths for the purpose of setting a local
host clock with respect to a remote reference while maintaining
accuracies in the order of ten milliseconds. The results demonstrate
the effectiveness of these algorithms in detecting and discarding
glitches due to hardware or software failure or operator mistakes.
They also demonstrate that time synchronization can be maintained
across the ARPANET in the order of ten milliseconds in spite of
glitches many times the mean roundtrip delay.

The results point to the need for an improved time-synchronization
protocol combining the best features of the ICMP Timestamp message
[6] and UDP Time protocol [7]. Among the features suggested for this
protocol are the following:

    1. The protocol should be based on UDP, which provides the
        flexibility to handle simultaneous, multiplexed queries and
        responses.

    2. The message format should be based on the ICMP Timestamp
        message format, which provides the arrival and departure times
        at the server and allows the client to calculate the roundtrip
        delay and offset accurately.

3. The data format should be based on the UDP Time format, which
    specifies 32-bit time in seconds since 1 January 1900, but
    extended additional bits for the fractional part of a second.

4. Provisions to specify the expected accuracy should be included
    along with information about the reference clock or
    synchronizing mechanism, as well as the expected drift rate
    and the last time the clock was set or synchronized.

The next step should be formulating an appropriate protocol with the
above features, together with implementation and test in the Internet
environment. Future development should result in a distributed,
symmetric protocol, similar perhaps to those described in [1], for
distributing highly reliable timekeeping information using a
hierarchical set of trusted clocks.

6. References

1. Lindsay, W.C., and A.V. Kantak. Network synchronization of
    random signals. IEEE Trans. Comm. COM-28, 8 (August 1980),
    1260-1266.

2. Mills, D.L. Time Synchronization in DCNET Hosts. DARPA Internet
    Project Report IEN-173, COMSAT Laboratories, February 1981.

3. Mills, D.L. DCNET Internet Clock Service. DARPA Network Working
    Group Report RFC-778, COMSAT Laboratories, April 1981.

4. Mills, D.L. Internet Delay Experiments. DARPA Network Working
    Group Report RFC-889, M/A-COM Linkabit, December 1983.

5. Mills, D.L. DCN Local-Network Protocols. DARPA Network Working
    Group Report RFC-891, M/A-COM Linkabit, December 1983.

6. Postel, J. Internet Control Message Protocol. DARPA Network
    Working Group Report RFC-792, USC Information Sciences Institute,
    September 1981.

7. Postel, J. Time Protocol. DARPA Network Working Group Report
    RFC-868, USC Information Sciences Institute, May 1983.

8. Postel, J. Daytime Protocol. DARPA Network Working Group Report
    RFC-867, USC Information Sciences Institute, May 1983.

9. Su, Z. A Specification of the Internet Protocol (IP) Timestamp
    Option. DARPA Network Working Group Report RFC-781. SRI
    International, May 1981.

     10. Marzullo, K., and S. Owicki. Maintaining the Time in a
         Distributed System. ACM Operating Systems Review 19, 3 (July
         1985), 44-54.

     11. Mills, D.L. Experiments in Network Clock Synchronization. DARPA
         Network Working Group Report RFC-957, M/A-COM Linkabit, September
         1985.

     12. Mills, D.L. Network Time Protocol (NTP). DARPA Network Working
         Group Report RFC-958, M/A-COM Linkabit, September 1985.

Appendix A.

   Experimental Determination of Internet Host Clock Accuracies

   Following is a summary of the results of three experiments designed
   to reveal the accuracies of various Internet host clocks. The first
   experiment uses the UDP Time protocol, which is limited in precision
   to one second, while the second uses the ICMP Timestamp message,
   which extends the precision to one millisecond. In the third
   experiment the results indicated by UDP and ICMP are compared. In
   the UDP Time protocol time is indicated as a 32-bit field in seconds
   past 0000 UT on 1 January 1900, while in the ICMP Timestamp message
   time is indicated as a 32-bit field in milliseconds past 0000 UT of
   each day.

   All experiments described herein were conducted from Internet host
   DCN6.ARPA, which is normally synchronized to a WWV radio clock. In
   order to improve accuracy during the experiments, the DCN6.ARPA host
   was resynchronized to a WWVB radio clock. As the result of several
   experiments with other hosts equipped with WWVB and WWV radio clocks
   and GOES satellite clocks, it is estimated that the maximum
   measurement error in the following experiments is less than about 30
   msec relative to standard NBS time determined at the Boulder/Fort
   Collins transmitting sites.

   A1. UDP Time Protocol Experiment

      In the first experiment four UDP Time protocol requests were sent
      at about three-second intervals to each of the 1775 hosts listed
      in the NIC Internet host table. A total of 555 samples were
      received from 163 hosts and compared with a local reference based
      on a WWVB radio clock, which is known to be accurate to within a
      few milliseconds. Not all of these hosts were listed as
      supporting the UDP Time protocol in the NIC Internet host table,
      while some that were listed as supporting this protocol either
      failed to respond or responded with various error messages.

In the following table "Host" is the canonical name of the host
and "Count" the number of replies received. The remaining data
represent the time offset, in seconds, necessary to correct the
local (reference) clock to agree with the host cited. The "Max"
and "Min" represent the maximum and minimum of these offsets,
while "Mean" represents the mean value and "Var" the variance, all
rounded to the nearest second.

Host	Count	Max	Min	Mean	Var
BBN-CLXX.ARPA	4	-11	-12	-11	0
BBN-KIWI.ARPA	4	-11	-12	-11	0
BBN-META.ARPA	4	-11	-12	-11	0
BBNA.ARPA	1	22	22	22	0
BBNG.ARPA	4	87	87	87	0
BELLCORE-CS-GW.ARPA	3	72	71	71	0
BLAYS.PURDUE.EDU	2	-1	-1	-1	0
CMU-CC-TE.ARPA	4	-94	-95	-94	0
CMU-CS-C.ARPA	3	6	5	5	0
CMU-CS-CAD.ARPA	4	-37	-37	-37	0
CMU-CS-CFS.ARPA	3	-42	-43	-42	0
CMU-CS-G.ARPA	3	-30	-31	-30	0
CMU-CS-GANDALF.ARPA	3	-42	-43	-42	0
CMU-CS-H.ARPA	4	-36	-37	-36	0
CMU-CS-IUS.ARPA	3	-44	-45	-44	0
CMU-CS-IUS2.ARPA	3	-44	-44	-44	0
CMU-CS-K.ARPA	3	-31	-31	-31	0
CMU-CS-SAM.ARPA	4	-74	-75	-74	0
CMU-CS-SPEECH.ARPA	4	-39	-40	-39	0
CMU-CS-SPEECH2.ARPA	4	-49	-50	-49	0
CMU-CS-SPICE.ARPA	4	-131	-132	-131	0
CMU-CS-THEORY.ARPA	4	-36	-37	-36	0
CMU-CS-UNH.ARPA	4	-44	-45	-44	0
CMU-CS-VLSI.ARPA	4	-66	-66	-66	0
CMU-RI-ARM.ARPA	3	-41	-41	-41	0
CMU-RI-CIVE.ARPA	3	-44	-45	-44	0
CMU-RI-FAS.ARPA	4	-27	-28	-27	0
CMU-RI-ISL1.ARPA	4	-18	-19	-18	0
CMU-RI-ISL3.ARPA	3	-49	-50	-49	0
CMU-RI-LEG.ARPA	3	-33	-33	-33	0
CMU-RI-ML.ARPA	4	42	42	42	0
CMU-RI-ROVER.ARPA	4	-48	-49	-48	0
CMU-RI-SENSOR.ARPA	2	-40	-41	-40	0
CMU-RI-VI.ARPA	3	-65	-65	-65	0
COLUMBIA.ARPA	1	8	8	8	0
CU-ARPA.CS.CORNELL.EDU	4	5	3	4	0
CYPRESS.ARPA	4	2	1	1	0

DCN1.ARPA	4	0	0	0	0
DCN5.ARPA	4	0	0	0	0
DCN6.ARPA	4	0	0	0	0
DCN7.ARPA	4	-1	-1	0	0
DCN9.ARPA	4	-3	-3	-3	0
DEVVAX.TN.CORNELL.EDU	2	3659	3658	3658	0
ENEEVAX.ARPA	4	73	72	72	0
FORD-WDL1.ARPA	4	-59	-60	-59	0
FORD1.ARPA	4	0	0	0	0
GUENEVERE.PURDUE.EDU	3	1	0	0	0
GVAX.CS.CORNELL.EDU	4	-18	-18	-18	0
IM4U.ARPA	4	-6	-6	-6	0
IPTO-FAX.ARPA	4	0	0	0	0
KANKIN.ARPA	4	-3	-4	-3	0
MERLIN.PURDUE.EDU	2	3	3	3	0
MIT-ACHILLES.ARPA	4	16	16	16	0
MIT-AGAMEMNON.ARPA	4	-63	-64	-63	0
MIT-ANDROMACHE.ARPA	4	-28	-28	-28	0
MIT-APHRODITE.ARPA	4	-7	-8	-7	0
MIT-APOLLO.ARPA	4	-8	-9	-8	0
MIT-ARES.ARPA	4	-25	-26	-25	0
MIT-ARTEMIS.ARPA	4	-34	-35	-34	0
MIT-ATHENA.ARPA	4	-37	-37	-37	0
MIT-ATLAS.ARPA	4	-26	-26	-26	0
MIT-CASTOR.ARPA	4	-35	-35	-35	0
MIT-DAFFY-DUCK.ARPA	2	-72	-73	-72	0
MIT-DEMETER.ARPA	4	-28	-29	-28	0
MIT-GOLDILOCKS.ARPA	1	-20	-20	-20	0
MIT-HECTOR.ARPA	4	-23	-24	-23	0
MIT-HELEN.ARPA	4	6	5	5	0
MIT-HERA.ARPA	4	-34	-35	-34	0
MIT-HERACLES.ARPA	4	-36	-36	-36	0
MIT-JASON.ARPA	4	-36	-37	-36	0
MIT-MENELAUS.ARPA	4	-32	-33	-32	0
MIT-MULTICS.ARPA	3	25	23	24	0
MIT-ODYSSEUS.ARPA	4	20	19	19	0
MIT-ORPHEUS.ARPA	4	-34	-35	-34	0
MIT-PARIS.ARPA	4	-35	-35	-35	0
MIT-POSEIDON.ARPA	4	-39	-41	-40	0
MIT-PRIAM.ARPA	4	-24	-25	-24	0
MIT-REAGAN.ARPA	4	115	115	115	0
MIT-THESEUS.ARPA	4	-43	-44	-43	0
MIT-TRILLIAN.ARPA	4	-38	-39	-38	0
MIT-TWEETY-PIE.ARPA	3	106	105	105	0
MIT-ZERMATT.ARPA	4	-75	-76	-75	0
MIT-ZEUS.ARPA	4	-37	-39	-38	0
MOL.ARPA	2	-3	-3	-3	0

MUNGO.THINK.COM	4	-1	-1	-1	0
NETWOLF.ARPA	4	158	157	157	0
ORBIT.ARPA	3	-4	-5	-4	0
OSLO-VAX.ARPA	3	3729	3727	3728	1
PATCH.ARPA	1	18	18	18	0
RADC-MULTICS.ARPA	4	-14	-15	-14	0
RICE-ZETA.ARPA	1	-31	-31	-31	0
RICE.ARPA	1	7	7	7	0
ROCHESTER.ARPA	4	-18	-18	-18	0
ROCK.THINK.COM	4	2	2	2	0
SCRC-QUABBIN.ARPA	4	-100	-100	-100	0
SCRC-RIVERSIDE.ARPA	4	-128	-128	-128	0
SCRC-STONY-BROOK.ARPA	4	-100	-100	-100	0
SCRC-VALLECITO.ARPA	4	-57	-57	-57	0
SCRC-YUKON.ARPA	4	-65	-65	-65	0
SEBASTIAN.THINK.COM	4	-14	-15	-14	0
SEISMO.CSS.GOV	3	-1	-1	0	0
SRI-BISHOP.ARPA	4	-40	-41	-40	0
SRI-DARWIN.ARPA	2	-29	-30	-29	0
SRI-HUXLEY.ARPA	2	-28	-29	-28	0
SRI-KIOWA.ARPA	4	-29	-30	-29	0
SRI-LASSEN.ARPA	3	-11	-12	-11	0
SRI-MENDEL.ARPA	4	74	73	73	0
SRI-PINCUSHION.ARPA	4	-50	-51	-50	0
SRI-RITTER.ARPA	4	-23	-24	-23	0
SRI-TIOGA.ARPA	4	127	127	127	0
SRI-UNICORN.ARPA	4	-38486	-38486	-38486	0
SRI-WHITNEY.ARPA	4	-24	-24	-24	0
SRI-YOSEMITE.ARPA	4	-26	-27	-26	0
SU-AIMVAX.ARPA	2	-54	-55	-54	0
SU-COYOTE.ARPA	1	14	14	14	0
SU-CSLI.ARPA	4	-1	-1	-1	0
SU-PSYCH.ARPA	1	-52	-52	-52	0
SU-SAFE.ARPA	1	-60	-60	-60	0
SU-SIERRA.ARPA	4	-53	-53	-53	0
SU-SUSHI.ARPA	4	-105	-106	-105	0
SU-WHITNEY.ARPA	2	-14	-14	-14	0
TESLA.EE.CORNELL.EDU	3	-2	-3	-2	0
THORLAC.THINK.COM	4	-20	-20	-20	0
TRANTOR.ARPA	4	4	3	3	0
TZEC.ARPA	4	-6	-7	-6	0
UBALDO.THINK.COM	4	-13	-13	-13	0
UCI-CIP.ARPA	2	-566	-567	-566	0
UCI-CIP2.ARPA	2	-175	-175	-175	0
UCI-CIP3.ARPA	2	-89	-90	-89	0
UCI-CIP4.ARPA	2	-51	-51	-51	0
UCI-CIP5.ARPA	2	-26	-28	-27	1

UCI-ICSA.ARPA	2	-24	-24	-24	0
UCI-ICSC.ARPA	1	0	0	0	0
UCI-ICSD.ARPA	1	-24	-24	-24	0
UCI-ICSE.ARPA	1	-10	-10	-10	0
UDEL-DEWEY.ARPA	1	88	88	88	0
UDEL-MICRO.ARPA	2	64	64	64	0
UIUC.ARPA	4	105	103	104	0
UIUCDCSB.ARPA	4	65	65	65	0
UMD1.ARPA	4	0	0	0	0
UMICH1.ARPA	4	-1	-1	0	0
UO.ARPA	4	-2	-3	-2	0
USC-ISI.ARPA	4	-45	-45	-45	0
USC-ISIC.ARPA	4	28	26	27	0
USC-ISID.ARPA	4	26	25	25	0
USC-ISIE.ARPA	4	-53	-54	-53	0
USC-ISIF.ARPA	4	-29	-29	-29	0
USGS2-MULTICS.ARPA	3	75	74	74	0
UT-ALAMO.ARPA	4	22	22	22	0
UT-BARKLEY.ARPA	4	57	56	56	0
UT-EMIL.ARPA	4	29	28	28	0
UT-GOTTLOB.ARPA	4	42	41	41	0
UT-HASKELL.ARPA	4	6	6	6	0
UT-JACQUES.ARPA	4	21	20	20	0
UT-SALLY.ARPA	3	1	0	0	0
VALENTINE.THINK.COM	4	-10	-11	-10	0
WENCESLAS.THINK.COM	4	-2	-3	-2	0
XAVIER.THINK.COM	4	-14	-14	-14	0
XEROX.ARPA	4	0	0	0	0
YAXKIN.ARPA	3	-4	-5	-4	0
YON.THINK.COM	4	-11	-12	-11	0
ZAPHOD.PURDUE.EDU	4	-230	-231	-230	0
ZOTZ.ARPA	4	17	16	16	0

Table A1. UDP Host Clock Offsets for Various Internet Hosts

The above list includes several host clocks known to be
synchronized to various radio clocks, including DCN1.ARPA (WWVB),
DCN6.ARPA (WWV) and FORD1.ARPA (GOES). Under normal radio
receiving conditions these hosts should be accurate to well within
a second relative to NBS standard time. Certain other host clocks
are synchronized to one of these hosts using protocols described
in RFC-891, including DCN5.ARPA, DCN7.ARPA and UMD1.ARPA
(synchronized to DCN1.ARPA) and UMICH1.ARPA (synchronized to
FORD1.ARPA). It is highly likely, but not confirmed, that several
other hosts with low offsets derive local time from one of these
hosts or from other radio clocks.

The raw statistics computed from the weighted data indicate a mean
of -261 sec, together with a maximum of 3729 sec and a minimum of
-38486 sec. Obviously, setting a local clock on the basis of
these statistics alone would result in a gross error.

A closer look at the distribution of the data reveals some
interesting features. Table A2 is a histogram showing the
distribution within a few seconds of reference time. In this and
following tables, "Offset" is in seconds and indicates the
lower-valued corner of the histogram bin, which extends to the
next higher value, while "Count" indicates the number of samples
falling in that bin.

```
            Offset  Count            Offset  Count
            -------------            -------------
            0 sec   13               (continued)
            1       1               -1       3
            2       1               -2       3
            3       2               -3       3
            4       1               -4       2
            5       2               -5       0
            6       1               -6       2
            7       1               -7       1
            8       1               -8       1
            9       0               -9       0
            > 9     30              < -9     95
```

            Table A2. Offset Distribution < 9 sec

A total of 16 of the 163 host clocks are within a second in
accuracy, while a total of 125 are off more than ten seconds. It
is considered highly likely that most of the 16 host clocks within
a second in offset are synchronized directly or indirectly to a
radio clock. Table A3 is a histogram showing the distribution over
a larger scale.

```
              Offset  Count           Offset  Count
              -------------           -------------
              0 sec   35              (continued)
              30      3               -30     50
              60      8               -60     42
              90      3               -90     8
              120     1               -120    4
              150     1               -150    2
              180     0               -180    1
              210     0               -210    0
              240     0               -240    1
              270     0               -270    0
              > 270   2               < -270  2
```

          Table A3. Offset Distribution < 270 sec

   A total of 138 of the 163 host clocks are within a minute in
   accuracy, while a total of four host clocks are off more than 4.5
   minutes. It is considered likely that most host clocks, with the
   exception of the 16 identified above as probably synchronized to a
   radio clock, are set manually by an operator. Inspection of the
   raw data shows some hosts to be very far off; for instance,
   SRI-UNICORN.ARPA is off more than ten hours. Note the interesting
   skew in the data, which show that most host clocks are set slow
   relative to standard time.

   A2. ICMP Timestamp Messages Experiment

   The the second experiment four ICMP Timestamp messages were sent
   at about three-second intervals to each of the 1775 hosts and 110
   gateways listed in the NIC Internet host table. A total of 1910
   samples were received from 504 hosts and gateways and compared
   with a local reference based on a WWVB radio clock, which is known
   to be accurate to within a few milliseconds. Support for the ICMP
   Timestamp messages is optional in the DoD Internet protocol suite,
   so it is not surprising that most hosts and gateways do not
   support it. Moreover, bugs are known to exist in several widely
   distributed implementations of this feature. The situation proved
   an interesting and useful robustness test for the clustering
   algorithm described in the main body of this note.

   While the complete table of ICMP offsets by host is too large to
   reproduce here, the following Tables A4 through A7 show the
   interesting characteristics of the distribution. The raw
   statistics computed from the weighted data indicate a mean of
   -2.8E+6 msec, together with a maximum of 8.6E+7 msec and a minimum
   of -8.6E+7 msec. Setting a local clock on the basis of these

statistics alone would be ridiculous; however, as described in the
main body of this note, use of the clustering algorithm improves
the estimate to within 8 msec of the correct value. The apparent
improvement of about six orders in magnitude is so remarkable as
to require a closer look at the distributions.

The reasons for the remarkable success of the clustering algorithm
are apparent from closer examination of the sequence of histograms
shown in Tables A4 through A7. Table A4 shows the distribution in
the scale of hours, from which it is evident that 80 percent of
the samples lie in a one-hour band either side of zero offset;
but, strangely enough, there is a significant dispersion in
samples outside of this band, especially in the negative region.
It is almost certain that most or all of the latter samples
represent defective ICMP Timestamp implementations. Note that
invalid timestamps and those with the high-order bit set
(indicating unknown or nonstandard time) have already been
excluded from these data.

```
          Offset  Count          Offset  Count
          -------------          -------------
          0 hr    204            (continued)
          1       10             -1      194
          2       0              -2      0
          3       0              -3      2
          4       0              -4      17
          5       0              -5      10
          6       0              -6      1
          7       0              -7      22
          8       0              -8      20
          9       0              -9      0
          > 9     0              < -9    13
```

             Table A4. ICMP Offset Distribution < 9 hours

Table A5 shows the distribution compressed to the range of 4.5
minutes. About half of the 370 samples remaining after the
outliers beyond 4.5 minutes are excluded lie in the band 30
seconds either side of zero offset, with a gradual tapering off to
the limits of the table. This type of distribution would be
expected in the case of host clocks set manually by an operator.

```
              Offset  Count              Offset  Count
              ------------               ------------
              0 sec   111                (continued)
              30      25                 -30     80
              60      26                 -60     28
              90      13                 -90     18
              120     7                  -120    19
              150     5                  -150    9
              180     3                  -180    10
              210     3                  -210    6
              240     1                  -240    2
              270     2                  -270    2
              > 270   29                 < -270  105
```

              Table A5. ICMP Offset Distribution < 270 sec

Table A6 shows the distribution compressed to the range of 27
seconds. About 29 percent of the 188 samples remaining after the
outliers beyond 27 seconds are excluded lie in the band 3 seconds
either side of zero offset, with a gradual but less pronounced
tapering off to the limits of the table. This type of
distribution is consistent with a transition region in which some
clocks are set manually and some by some kind of protocol
interaction with a reference clock. A fair number of the clocks
showing offsets in the 3-27 second range have probably been set
using the UDP Time protocol at some time in the past, but have
wandered away as the result of local-oscillator drifts.

```
              Offset  Count              Offset  Count
              ------------               ------------
              0 sec   32                 (continued)
              3       15                 -3      22
              6       9                  -6      12
              9       6                  -9      8
              12      13                 -12     8
              15      5                  -15     5
              18      8                  -18     9
              21      8                  -21     7
              24      9                  -24     3
              27      6                  -27     3
              > 27    114                < -27   202
```

              Table A6. ICMP Offset Distribution < 27 sec

Finally, Table A7 shows the distribution compressed to the range
of 0.9 second. Only 30 of the original 504 samples have survived
and only 12 of these are within a band 0.1 seconds either side of

zero offset. The latter include those clocks continuously
synchronized to a radio clock, such as the DCNet clocks, some
FORDnet and UMDnet clocks and certain others.

```
            Offset  Count              Offset  Count
            ------------              ------------
            0 sec   6                  (continued)
            .1      3                  -.1     6
            .2      1                  -.2     3
            .3      1                  -.3     0
            .4      0                  -.4     0
            .5      1                  -.5     2
            .6      0                  -.6     0
            .7      1                  -.7     0
            .8      4                  -.8     2
            .9      0                  -.9     0
            > .9    208                < -.9   266
```

            Table A7. ICMP Offset Distribution < .9 sec

   The most important observation that can be made about the above
   histograms is the pronounced central tendency in all of them, in
   spite of the scale varying over six orders of magnitude. Thus, a
   clustering algorithm which operates to discard outliers from the
   mean will reliably converge on a maximum-likelihood estimate close
   to the actual value.

   A3. Comparison of UDP and ICMP Time

   The third experiment was designed to assess the accuracies
   produced by the various host implementations of the UDP Time
   protocol and ICMP Timestamp messages. For each of the hosts
   responding to the UDP Time protocol in the first experiment a
   separate test was conducted using both UDP and ICMP in the same
   test, so as to minimize the effect of clock drift. Of the 162
   hosts responding to UDP requests, 45 also responded to ICMP
   requests with apparently correct time, but the remainder either
   responded with unknown or nonstandard ICMP time (29) or failed to
   respond to ICMP requests at all (88).

   Table A8 shows both the UDP time (seconds) and ICMP time
   (milliseconds) returned by each of the 45 hosts responding to both
   UDP and ICMP requests. The data are ordered first by indicated
   UDP offset and then by indicated ICMP offset. The seven hosts at
   the top of the table are continuously synchronized, directly or
   indirectly to a radio clock, as described earlier under the first

experiment. It is probable, but not confirmed, that those hosts
below showing discrepancies of a second or less are synchronized
on occasion to one of these hosts.

Host	UDP time	ICMP time
DCN6.ARPA	0 sec	0 msec
DCN7.ARPA	0	0
DCN1.ARPA	0	-6
DCN5.ARPA	0	-7
UMD1.ARPA	0	8
UMICH1.ARPA	0	-21
FORD1.ARPA	0	31
TESLA.EE.CORNELL.EDU	0	132
SEISMO.CSS.GOV	0	174
UT-SALLY.ARPA	-1	-240
CU-ARPA.CS.CORNELL.EDU	-1	-514
UCI-ICSE.ARPA	-1	-1896
UCI-ICSC.ARPA	1	2000
DCN9.ARPA	-7	-6610
TRANTOR.ARPA	10	10232
COLUMBIA.ARPA	11	12402
GVAX.CS.CORNELL.EDU	-12	-11988
UCI-CIP5.ARPA	-15	-17450
RADC-MULTICS.ARPA	-16	-16600
SU-WHITNEY.ARPA	17	17480
UCI-ICSD.ARPA	-20	-20045
SU-COYOTE.ARPA	21	21642
MIT-MULTICS.ARPA	27	28265
BBNA.ARPA	-34	-34199
UCI-ICSA.ARPA	-37	-36804
ROCHESTER.ARPA	-42	-41542
SU-AIMVAX.ARPA	-50	-49575
UCI-CIP4.ARPA	-57	-57060
SU-SAFE.ARPA	-59	-59212
SU-PSYCH.ARPA	-59	-58421
UDEL-MICRO.ARPA	62	63214
UIUCDCSB.ARPA	63	63865
BELLCORE-CS-GW.ARPA	71	71402
USGS2-MULTICS.ARPA	76	77018
BBNG.ARPA	81	81439
UDEL-DEWEY.ARPA	89	89283
UCI-CIP3.ARPA	-102	-102148
UIUC.ARPA	105	105843
UCI-CIP2.ARPA	-185	-185250
UCI-CIP.ARPA	-576	-576386
OSLO-VAX.ARPA	3738	3739395

```
        DEVVAX.TN.CORNELL.EDU   3657              3657026
        PATCH.ARPA             -86380             20411
        IPTO-FAX.ARPA          -86402             -1693
        NETWOLF.ARPA            10651435          -62164450
```

        Table A8. Comparison of UDP and ICMP Host Clock Offsets

Allowing for various degrees of truncation and roundoff abuse in
the various implementations, discrepancies of up to a second could
be expected between UDP and ICMP time. While the results for most
hosts confirm this, some discrepancies are far greater, which may
indicate defective implementations, excessive swapping delays and
so forth. For instance, the ICMP time indicated by UCI-CIP5.ARPA
is almost 2.5 seconds less than the UDP time.

Even though the UDP and ICMP times indicated by OSLO-VAX.ARPA and
DEVVAX.TN.CORNELL.EDU agree within nominals, the fact that they
are within a couple of minutes or so of exactly one hour early
(3600 seconds) lends weight to the conclusion they were
incorrectly set, probably by an operator who confused local summer
and standard time.

Something is clearly broken in the case of PATCH.ARPA,
IPTO-FAX.ARPA and NETWOLF.ARPA. Investigation of the PATCH.ARPA
and IPTO-FAX.ARPA reveals that these hosts were set by hand
accidently one day late (-86400 seconds), but otherwise close to
the correct time-of-day. Since the ICMP time rolls over at 2400
UT, the ICMP offset was within nominals. No explanation is
available for the obviously defective UDP and ICMP times indicated
by NETWOLF.ARPA, although it was operating within nominals at
least in the first experiment.