***I.J. Wireless and Microwave Technologies*, 2011, 4, 1-9**
Published Online August 2011 in MECS (http://www.mecs-press.net)
DOI: 10.5815/ijwmt.2011.04.01

Available online at http://www.mecs-press.net/ijwmt

Study on the Distribution of Networked Devices' Clock Skew

Jiao Chengbo[a,b], Zheng Hui[b]

[a] *Information Engineering Institute, Information Engineering University, Zhengzhou, Henan, China*
[b] *Southwestern Institute of Electronics and Telecommunication, Chengdu, Sichuan, China*

Abstract

Clock skews of devices on the Internet are viewed as one way delay noise, but their distribution is unknown. We explore the distribution of clock skews to see the conflict probability. In this paper, we introduce an accurate clock skew estimation algorithm to filter inaccurate clock skew estimation by comparing the results between linear programming method and least square fitting. Delay jitter and other noises affect the estimation result. When the difference of two methods is large, the estimation result is unstable and inaccurate, so the estimation result should be dropped. Based on this algorithm, we use traces of real Internet measurements to collect 1825 accurate clock skews of different devices to establish a fingerprint database. Furthermore, we show the distribution of clock skews and comparing conflict probability with different number of devices. The distribution shows that clock skews are diverse, and most of clock skews are in the region of [-100, 100] PPM. The results indicate that when the number of devices is small (<5), clock skews won't be conflict with each other, so clock skews are good tools to detect faked devices or NAT; When the number of devices increases, the conflict probability increases linearly, so clock skews of different devices can not distinguish each devices effectively.

Index Terms: Clock Skew; Distribution; Internet; Noise; Conflict Probability

1. Introduction

With advances in wired and wireless networks, networked systems are becoming increasingly prevalent. The dynamic nature of the Internet allows concealment of network identity in the course of data transmission. It is found that different clocks run at different frequencies, so there's relative clock skew between them [8]. Paxson et al. uses forward and reverse path measurements of delay between a pair of hosts to deal with relative offset and clock skew [6]. In [5], Moon et al. propose linear programming method (LPM) to estimate clock skew in delay measurements. Kohno et al. found that different personal computers and servers have different clock skews and they are stable under different scenarios, so it can be used to fingerprint different devices even they are behind NAT [1]. Suman et al. use Time Synchronization Function (TSF) timestamps in the IEEE 802.11 beacon/probe response messages sent by the wireless access point (AP) to estimate clock skew of different APs [2]. Results show that unauthorized AP have different clock skews compared with authorized wireless AP, so it can be used to detect unauthorized AP. Russ uses linear regression algorithm to statistically estimate clock skews to fingerprint devices [3]. Murdoch et al. use the temperature on clock skew as a classical covert channel to detect devices in anonymity systems [4]. It is well known that different devices can have different clock skews, but there's no result about the distribution of clock skews to evaluate their conflict probability.

The rest of this paper is organized as follows. In Section II, we firstly define the terms and model needed to describe clock skew. An automated inaccurate clock skew estimation filtering algorithm is given by comparing different estimation methods. In Section III we collect a large number of accurate clock skews to establish fingerprint database, and describe their distribution. Based on distribution, we analyze the conflict probability. In Section IV, test of the conflict probability with real traces are studied. We conclude the paper in Section V by summarizing our work.

2. Clock Skew Model and Estimation

2.1. Clock skew terminology and model

The resolution of a clock is its updating frequency, for example the resolution of 1us means that the clock updates its notion of time in 1e-06 second increments. The reciprocal of clock's resolution is the nominal frequency, defined as HZ. Offset is the difference between the time reported by the clock and the 'true' time at a particular moment. In this paper, we are concerned with the short-term frequency drift of clocks between the beginning of a network transfer and its end.

Fig. 1 illustrates the delay measurements between sending device S and receiving device R.

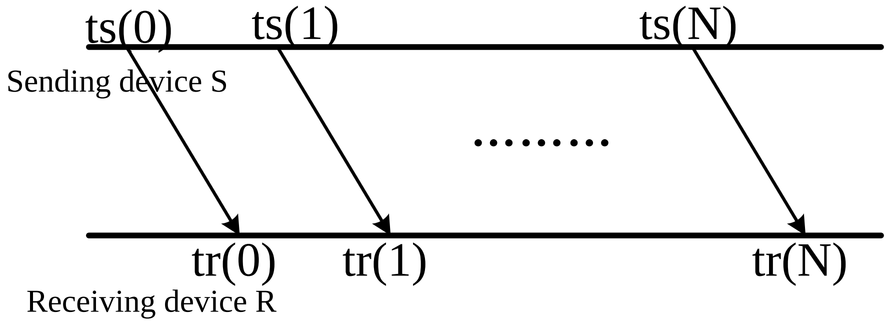

Fig. 1. One way delay measurements from S to R

Let $TS = \{ts_i, i \in [0, N]\}$ and $TR = \{tr_i, i \in [0, N]\}$ represent the leaving time and arriving time individually. The relative offset between S and R is

$$offset_{s->r}(i) = C_r(i) - C_s(i) \tag{1}$$

From (1), the relative offset between S and R can be calculated as:

$$OS = \{\{tr_i, (tr_i - tt_0) - (ts_i - ts_0), i \in [0, N]\} \tag{2}$$

System clock is a piecewise continuous function that is twice differentiable except on a finite set of points [5,8]. As clock skew is the difference in the frequencies between a clock and the 'true' clock, clock skew is defined as:

$$C_s'(t) = dC_s(t)/dt\, , C_r'(t) = dC_r(t)/dt \tag{3}$$

Let $\mathrm{var}_i = (tt_i - tt_0) - (ts_i - ts_0)$, from (2) and (3), the relative clock skew $CS_{s->r}$ is the slope of OS

$$CS_{s->r} = d\,\mathrm{var}_i / dtt_i \tag{4}$$

In the section B, different clock skew estimation algorithms are compared.

2.2. Automated inaccurate clock skew filtering method

Different methods are proposed for estimating the clock skew from the offset-set including Paxson's method, linear regression algorithm, piecewise minimum algorithm, LPM and least square fitting (LSF) [2,3,4,5,6,7]. Compared with LPM, Paxson's method, linear regression algorithm and piecewise minimum algorithm are more sensitive to the variability in data transmission. LPM remains stable even if there are significant number of outliers.

Let α_{lpm} and β_{lpm} are the slope and y-axis intercept of LPM, then

$$\begin{cases} \mathrm{var}_i \ge \alpha_{lpm}.Rr_i + \beta_{lpm} \\ \min\{1/N \sum_{i=1}^{N} (\mathrm{var}_i - \alpha_{lpm}.Rr_i - \beta_{lpm})\} \\ \mathrm{var}_i = (tt_i - tt_0) - (ts_i - ts_0) \\ Rr_i = tr_i - tr_0 \end{cases} \tag{5}$$

From (5), compared with other methods, LPM fit a line that lies under all the offset data points as closely as possible, so it's more accurate for the same data set. But when the portion of outliers is large caused by delay jitter and other noises, α_{lpm} won't be accurate.

Let α_{lsf} and β_{lsf} are the slope and y-axis intercept of LSF, then

$$\min\{\sum_{i=1}^{N} (\mathrm{var}_i - (\alpha_{lsf}.Rr_i - \beta_{lsf})^2)\} \tag{6}$$

Compared with LPM, LSF is sensitive to outliers, even a small number of outliers will affect α_{lsf}. But when the portion of outliers is not very large, LSF will give an accurate value determined by the majority of the offset-set points. Our idea is that LSF'S sensitivity to the presence of even a small number of outliers will help determining the accuracy of LPM. Fig. 2 and 3 illustrate the difference between LPM and LSF in high and low portion of outliers. Let d is the difference between α_{lsf} and α_{lpm}, so it is calculated as

$$d = |\alpha_{lsf} - \alpha_{lpm}| \tag{7}$$

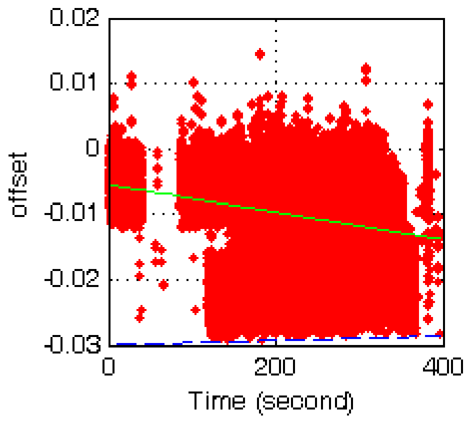

Fig. 2. High portion of outliers

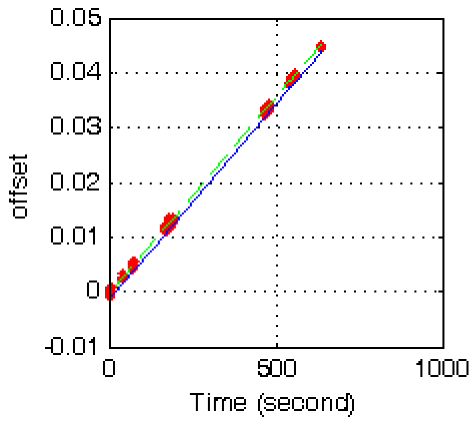

Fig. 3. Low portion of outliers

Table I gives the comparison results.

Table 1. Comparison of LPM and LSF

Outlier portion	LPM (PPM)	LSF (PPM)	Difference (PPM)
high	3.4	-20.8	24.2
low	71.4	71.8	0.4

In figure 2, although the last time is more than 400 seconds, but the delay jitters lead to large d and unstable α_{lpm}. In figure 3, the difference d is small (<1 PPM), and α_{lsf} is stable. Then we can use d to test the validity of estimated clock skew to filter inaccurate results. ICSFilter (Inaccurate Clock Skew Filter) algorithm is proposed to automated filter inaccurate clock skew as follows:

```
Input: $O_N = \{(Rr_i, \text{var}_i), i \in [1, N]\}$ and threshold $pv$
    Output: $CS_{s->r}$
        Begin
$\alpha_{lpm}$ =lsf({($Rr_1, \text{var}_1$),..., ($Rr_N, \text{var}_N$)})
$\alpha_{lsf}$ =lsf({($Rt_1, \text{var}_1$),..., ($Rr_N, \text{var}_N$)})
        End
if(abs($\alpha_{lpm}$ - $\alpha_{lsf}$) $\leq pv$)
$CS_{s->r} = \alpha_{lpm}$
else
        $CS_{s->r} = NULL$
```

3. Distribution of Clock Skew

Based on ICSFilter algorithm in Section II, 1825 accurate clock skews are collected and recorded into database for distribution analysis in this section.

3.1. Establishing clock skew database

There's no NAT device in the LAN (Local Area Network), and each device has a relative constant IP address, so IP address can be used as the ID to represent device. When capturing packet, TCP timestamp and arriving time are extracted or recorded to form a data node.

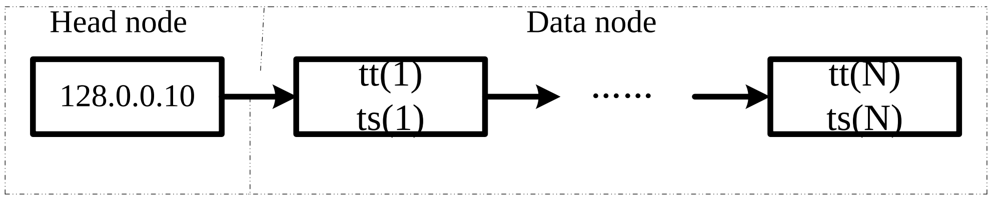

Fig. 4. Record of a source IP address

When new packet arrives, the packet number parameter $PktNum$ and duration parameter Dur are updated to check whether they have met the threshold. When they meet the threshold, clock skew is estimated based on ICSFilter to check for validity. In this paper, $PktNum$ is set as 500, and Dur is set as 700 seconds. After collecting for more than 1 week, 1825 clock skew of different devices are recorded into database.

3.2. Clock skew distribution with different operating systems

The devices in the LAN run a wide variety of operating systems and hardware, in order to see whether distribution of clock skews will be affected by clock resolution, we give the distribution statistics of clock skews with different operating systems and hardware. Windows (XP/7) and Linux (kernel 2.4/2.6) are the main two operating systems running in the LAN. Their distribution statistics (probability density function, PDF) are illustrated in figure 5 and 6.

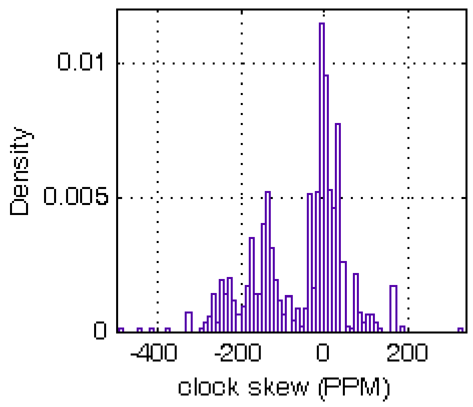

Fig. 5. Clock skews (Windows) distribution statistics

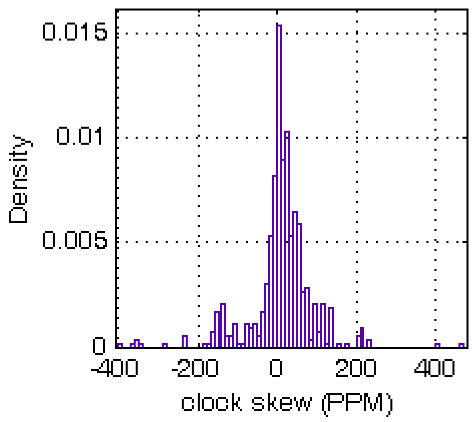

Fig. 6. Clock skews (Linux) distribution statistics

It is found that the two distribution statistics are similar with each other, so clock skew distribution statistics is not affected by systems.

3.3. Distribution

In this experiment, we analyzed clock skew distribution of a variety of machines including computers, notebooks and printers with different operating systems. More than 95% results are ranged in [-500, 500] PPM, the probability density function are showed in Figure 7.

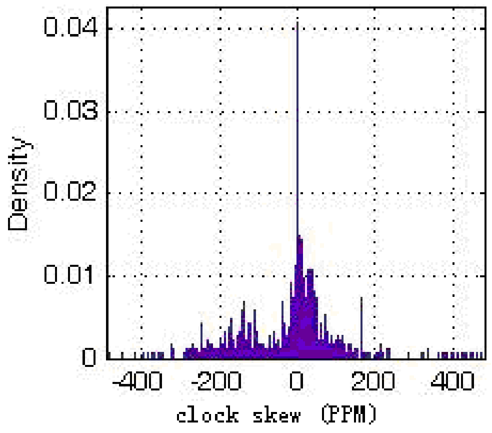

Fig. 7. Clock skews distribution statistics

The largest probability is 2.6% which is in the region of [0, 1] PPM, and more than 70% of clock skews focus in the region of [-100, 100] PPM. Figure 8 shows the distribution restricted in the region of [-100, 100] PPM.

From Figure 8, it is illustrated that clock skews' distribution restricted in [-100, 100] PPM fits normal distribution ($\mu = 10.2$ and $\sigma = 34$), which indicates that more than 95% clock skews (restricted in [-100, 100] PPM) focus in [-57, 78] PPM. In the next Section, we will analyze the conflict probability based on distribution result.

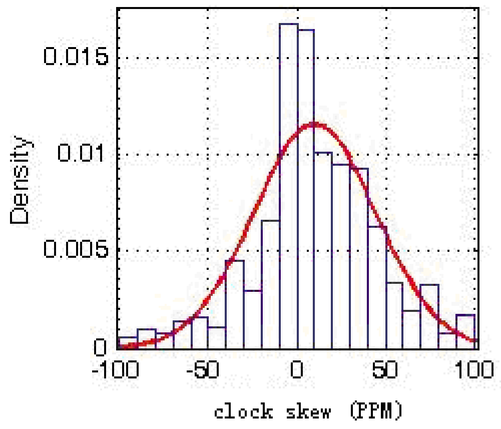

Fig. 8. Clock skews (restricted in [-100, 100] PPM) distribution statistics

4. Experiment

From figure 7, it is found that $N = 1000$. The differentiation threshold is set as $pv = 1$ PPM. In this experiment, clock skews of n devices are randomly selected to calculate conflict probability. For each of clock skews, if they are within pv PPM, then conflict occurs.

$$\exists i, j \in [1, n], i \neq j, abs(CS_i - CS_j) < pv \tag{11}$$

For each n ($n \in [2,10]$), 50 tests are run. Figure 9 shows the result. The result is consistent with analysis in conflict probability. It is found that when n is small (<5), conflict probability is very low. So faked devices can be easily detected by clock skew. And if there're more than one device behind NAT, their clock skews will differ with each other significantly. With the increase of n, the conflict probability increases linearly, so conflict occurs easily. Then devices won't be accurately distinguished only with clock skews when n is more than 5.

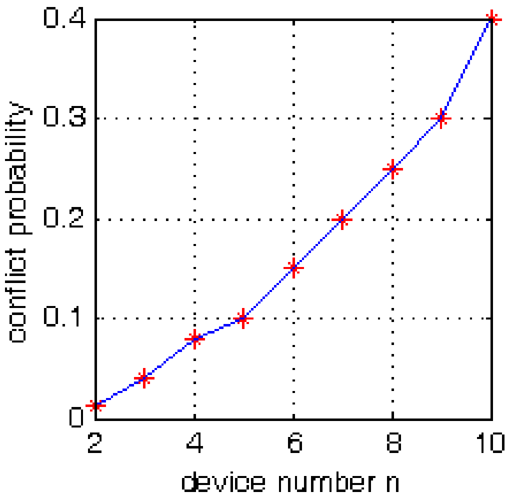

Fig. 9. Conflict probability

5. Conclusions

In this paper, we developed an automated inaccurate clock skew filtering methodology to guarantee accuracy of estimated clock skew records. Applying this method to massive data set, 1825 accurate clock skews are estimated to establish the clock skew database. Distribution of clock skew running on different systems shows that system won't affect clock skew distribution. And more than 70% clock skews focus in [-100, 100] PPM which fits normal distribution. Analysis and test of conflict probability show that when the number of devices is small, conflict probability is very low, but linearly increase with number. The result shows that clock skew can be used to detect faked device or NAT effectively. But when devices increate it can't distinguish devices accurately.

References

[1] T. Konoho, A. Broido, and K. Claffy. Remote physical device fingerprinting. In Proceedings of IEEE Symposium on Security and Privacy, Oakland, California, May 2005.
[2] J. Suman, K. Sneha. On fast and accurate detection of unauthorized wireless access points using clock skews. In the ACM Sigmobile International Conference on Mobile Computing and Networking, San Francisco, California, USA ,2008.
[3] F. RUSS. A statistical approach to remote physical device fingerprinting. In Proc. of the military communications conference, Orlando, USA, October 2007.
[4] S. Murdoch， Hot or Not: Revealing Hidden Service by their clock skew. In Proc of the 13th ACM Conference on Computer and Communications Security. Alexandra, VA, USA, 2006.
[5] S. B. Moon, P Skelly.; Towsley, D. Estimation and removal of clock skew from network delay measurements. In Proc of the INFOCOM, New York, USA, March 1999.
[6] V. Paxson, Measurements and Analysis of End-to-End Internet Dynamics, Ph. D. thesis, University of California, Berkeley, 1997.
[7] V. Paxson, 'On calibrating measurements of packet transit times', in Proceedings of SIGMETRICS'98, Madison, Wisconsin, June, 1998.
[8] D. Mills. Network time protocol (Version 3) specification, implementation and analysis. RFC 1305. March 1992.