Solving Time Synchronization in Windows-Based Networks

White paper

Microsemi
Power Matters.™

Microsemi Corporate Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA
Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Sales: +1 (949) 380-6136
Fax: +1 (949) 215-4996
E-mail: sales.support@microsemi.com
www.microsemi.com

About Microsemi

Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for aerospace & defense, communications, data center and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; enterprise storage and communication solutions, security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, Calif., and has approximately 4,800 employees globally. Learn more at www.microsemi.com.

Solving Time Synchronization in Windows-Based Networks

The Windows W32Time service is not designed to provide the sub-millisecond accuracy that is required in many modern use cases. The solution? Microsemi's Domain Time II.

Many IT managers are surprised to learn that their Windows networks are considered out of sync, based both on their use cases and on the two-part definition of "correct" time: accuracy and precision. Computer clocks are accurate when they sync with the actual time-of-day and they are precise when they sync with each other. However, most Windows computers are out of sync by seconds or even minutes, and do not fully support the industry's network time protocol (NTP) designed to keep computers in sync.

Microsoft refers to this as "loose sync," and it can have consequences for how organizations use Windows in regulatory compliance, virtual computing, distributed computing, computer forensics, and legal record-keeping. An organization can decide that some use cases are not feasible or are significantly compromised, but in order to handle situations worth the extra effort, organizations can deploy a different computing platform (such as Linux) at the expense of deploying and maintaining another platform.

The ideal option is to replace the Windows W32Time service, which enables organizations to ensure more stringent timekeeping requirements for those use cases that would benefit from better timekeeping and monitoring.

Windows Timekeeping Must Be More Rigorous

Due to the proliferation of more powerful computers and cloud networks, organizations expect that their users can do more with their machines. One trend is that critical tasks are now more widely dispersed throughout the organization. Thus, tracking desktop systems, moment-by-moment activity for forensic, compliance, and security purposes are now of great importance. Another trend is that desktops may now perform tasks that previously required Linux servers (this includes tasks that relied on Linux servers' superior timekeeping). The result of both trends is that what used to be "good enough" timekeeping in a Windows environment no longer is acceptable.

Today's desktop timekeeping requirements are more rigorous in the following six areas.

- Accuracy: The accuracy of a computer clock is its offset from Universal Coordinated Time (UTC), typically measured in milliseconds.
- Precision: The precision of a computer clock is the rate at which it ticks relative to other clocks (in the frequency domain) or the interval between ticks (in the time domain) relative to other clocks.
- Scalability: As the demands on computers from various applications grow, so too does the need for accuracy relative to UTC when logging the associated data.
- Reliability: If applications rely on good timekeeping, then the system timekeeping must be reliable.
- Ease-of-management: The easier it is to deploy and administer precise and accurate time synchronization across the network, the more likely it is that network timekeeping will be precise and accurate.
- Security: One way to corrupt or take down applications on a network is to degrade its timing infrastructure with denial of service (DoS) or other attacks.

What's different now is the extent to which these criteria matter due to the changing mix of Windows applications and the requirements they impose.

Why the Active Directory Needs Time

Active Directory provides a set of services to Windows domain networks, including:

- Authentication for all users and computers
- Enforcement of security policies for all computers
- Software updates
- Enforcement of roles (for example, user or administrator) at login

In early versions of Windows (for example, Windows NT 3.51 and 4.0), the task of processing all updates (such as authorizing a new user) for a single domain was handled by the Primary Domain Controller (PDC). In later versions of Windows, the Flexible Single Master Operation (FSMO) emulates the PDC so the FSMO is not bound to a single domain controller.

One of the tasks of PDC emulator is to synchronize time as required by the Kerberos authentication protocol. The time service ensures a hierarchical relationship in control of authority and prohibits loops. Each PDC emulator role holder in a Windows domain is responsible for acquiring time from its upstream "time partner" and distributing time to its downstream partner(s) in the hierarchy. The PDC emulator at the top of the hierarchy has ultimate authority and should be configured to acquire time from an external source.

The following illustration shows the Windows W32Time service hierarchy.

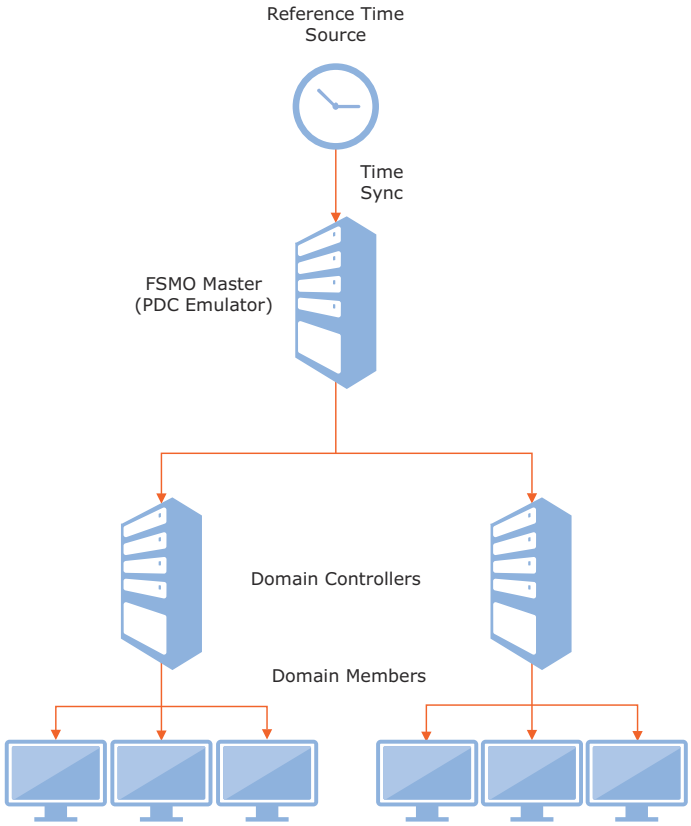

***Figure 1* • Windows W32Time Service Hierarchy**

Why Not Just Use the Computer's Clock?

The clocks that users see on their PCs are notorious for drifting. They are typically based on inexpensive oscillator circuits and can easily drift seconds to minutes per day, accumulating significant errors over time. That is why they constantly need to be reset, which occurs when the operating system requests time from an authorized source such as an NTP server.

Windows Applications have Become More Sensitive to Time

Better timekeeping has become a bigger issue on Windows in five key areas: compliance, virtual computing, distributed computing, forensics, and legal record-keeping. The following sections provide examples of the important role timekeeping plays in these areas.

Compliance

Many government regulations specify timekeeping requirements. One example is the Defense Federal Acquisition Regulation Supplement (DFARS), a set of regulations that government acquisition officials and DoD contractors must follow in the procurement of goods and services. Effective as of December 31, 2017, DFARS mandates that if an organization manages information subject to cyber incident reporting, then its information system must synchronize clocks to an authoritative time source in order to generate time stamps for audit records.

Another example is Europe's Markets in Financial Instruments Directive 2 (MiFID 2), which goes into effect from January 2018. The directive mandates that high-frequency trading systems must be synced to within 100 microseconds of UTC with 1-microsecond precision granularity. For non-human intervention trading, the mandate is 1 millisecond of UTC. Modern Windows operating systems can support sub-microsecond timing and most Windows versions supports sub-millisecond-range time, provided they have a sufficiently capable Windows time synchronization solution.

Other compliance examples include the Health Insurance Portability and Accountability Act of 1996 (HIPAA), Sarbanes-Oxley, and 21 CFR Part 11. The FDA's requirements for using electronic records and signatures all mandate specific requirements regarding the accuracy of time stamps on records.

Virtual Computing

Unlike applications that run on physical machines and have dedicated access to host resources (including interrupts and timers), applications running on Virtual Machines (each with its own W32Time service) share host resources. The lack of dedicated or deterministic hardware counters can significantly magnify Windows W32Time service errors, compromising performance in the other four areas discussed here.

Distributed Computing

Timekeeping is a significant performance issue for applications that employ a distributed computing model, such as a micro services-based architecture or service-oriented architecture. Unlike monolithic applications where all code is executed in a single run time, distributed applications are comprised of discrete self-contained services that interact with each other through Application Programming Interfaces (APIs). These services can run on the same machine, different machines in a cluster, or even in geographically dispersed machines. The major strengths of distributed computing is that each service can be highly optimized (for example, by running it on the best available hardware, by programming it using the most qualified team, or by using the most appropriate programming language). One of the weaknesses of distributed computing is its vulnerability to timing issues. If services are out of sync by even a few hundred milliseconds, the application may falter. Furthermore, monitoring tools designed to diagnose faults must be synchronized at even finer levels of granularity than the services they are monitoring in order to detect timing problems and to show along a consistent timeline when services were called and executed.

Forensics

Accurate and precise timekeeping also plays a significant role in ordering events and troubleshooting root-cause problems. By definition, all computer networks are comprised of distributed resources that interact in order for applications to run properly. When an error occurs due to either technical glitches or cyber-attacks, network administrators examine server log files to investigate the cause and effect of various events organized by their time stamps. That includes events related to firewall and VPN security-related activity, bandwidth usage, logging, management, authentication, authorization, and accounting functions.

Because server logs are a compilation of information from different hosts, it is essential that all the time stamps in these hosts' respective log files be synced so their data (such as centrally logged configuration events and system error messages, router configuration changes, interface up/down status, security alerts, environmental conditions, trace backs, and CPU process overloads) can be analyzed on the same timeline.

Legal Record-keeping

Time stamps on emails, corporate documents, and other electronic evidence have become an increasingly important part of legal proceedings to determine when and in what order events occurred. Accurate and verifiable time stamps are therefore essential to establish the traceability of events and the admissibility of evidence.

These use cases are good examples of when organizations need quality timekeeping, regardless of which operating systems they run. However, they are also good use cases for using Windows. So, what's wrong with the Windows time service?

W32Time Only Provides Loose Timing

According to Microsoft, the W32Time service was not designed to be a full-featured NTP solution for synchronizing nodes on a network. With a target accuracy of only two minutes, it is not feasible for many time critical applications where the time sync is in seconds or microseconds range. So why does Windows include W32Time in the first place? The purpose of W32Time is to enable Kerberos Version 5 authentication. Kerberos is a security protocol that prevents Windows clients from logging to a Windows server if the client violates certain rules. One of those rules is that the time on the client must agree with the time on the server (within certain limits).

W32Time is software that runs on clients and servers in a network hierarchy (see Figure 1). Servers reference time from an external source, such as a GPS receiver or a time server found on the Internet. Clients and downstream servers receive the time by sending requests that are authenticated and acknowledged by upstream servers through a W32Time's NTP variant. Kerberos then checks all incoming client packets to see if their time stamps agree (approximately) with the server's clock. If not, the packets are dropped.

As a time service, W32Time is bare bones. For example, setup and configuration are done through a basic command line interface, so adjustments to time can take up to 10 hours to propagate. Furthermore, no alerts are provided if issues (such as loss of a referenced time source) occur, and there is no auto failover (where W32Time instances take over for each other if one fails).

Of course, the quality of the time available from any time service is only as good as the referenced time source. Because W32Time timing accuracy is only measured in seconds at best, many organizations see little advantage in investing in an NTP server capable of millisecond performance, and so reference some NTP server found on the Internet. This can pose significant problems. First, such NTP servers may be unreliable, unsecured, and are commonly exposed to DoS attacks. Second, there can also be asymmetric propagation delays from when the timing packet is sent until it is received, causing the NTP time offset calculation to be significantly different from the server. Finally, NTP packets coming from outside the firewall can expose the network to threats such as packet spoofing or DoS attacks.

To mitigate these risks to their Windows environment, organizations often choose to replace their W32Time service with one built to support today's more rigorous timekeeping requirements for accuracy, precision, scalability, reliability, manageability, and security.

Microsemi's Domain Time II: A W32Time Replacement Model

A W32Time replacement model, such as Microsemi's Domain Time II, provides a real-world solution to many of the problems posed by W32Time's shortcomings.

Microsemi's Domain Time II is:

- Accurate and precise- first and foremost, a model for W32Time replacement service must hold network timing within the limits specified for Windows applications requiring millisecond performance (see Table 1). In addition, it can perform very fast, smooth, and precise time correction without jitter.
- Scalable- the service can distribute accurate and precise timing on any size network. It can also manage time synchronization as easily on large networks as small networks.
- Reliable- on large Windows domain networks, the software servers automatically take over for each other when one becomes unavailable, and clients automatically find alternate servers if a failure occurs. Time components can also be set manually for multiple levels of fallback time sources.
- Easy-to-manage- the replacement time service can replace W32Time in existing Windows time hierarchies. The service runs as an automatic background system service and does not replace any system executables or .dlls. The resource impact is very low and the footprint is small; administrators simply turn off W32Time and install the new software on all clients and servers from a single workstation (using a convenient GUI as opposed to a command line interface). The time hierarchy automatically adjusts to changes in the network, assuring that clients have continuing access to the correct time. Additional features include detailed event logging, alerts if set thresholds are exceeded, and complete records of time sync accuracy on the network.
- Secure- to protect against fraud, users and other programs/services cannot change their own time. Symmetric key authentication and Windows authentication are also supported.

The following table provides details of the W32Time and Domain Time II benchmarks.

Table 1 • **W32Time and Domain Time II Benchmarks**

Operating system	Protocol used in Domain Time II	Syntonization	Synchronization to the reference GPS server	Excursions
Win8.x, Server 2012 Win10, Server 2016	PTPv2	50 µs/sec	2 µs	Rare
Server 2003, XP	PTPv2	200 µs/sec	>50 µs	Occasional
Vista, Server 2008, Win7	PTPv2	300 µs/sec	>100 µs	Regular
Server 2003, XP, Win8.x, Server 2012, Win10, Server 2016	DT2 or NTP	1 ms/sec	>500 µs	Occasional
Vista, Server 2008, Win7	DT2 or NTP	100 ms/sec–1000 ms/sec	1 ms–100 ms	Excessive

Of course, you can't distribute what you can't acquire. That's why a strong Windows time distribution service is incomplete without a sufficiently capable timing source. For example, a GPS-referenced NTP server that delivers accurate and precise time even if GPS is unavailable will be under heavy load because of responding to many client timing requests, and can become a target for DoS attacks.

In comparison, consider the Microsemi S600 NTP Time Server. It offers hardware time stamping capable of handling 10,000 NTP requests per second with aggregate limiting of inbound packets across all ports to mitigate DoS attacks. Its optional NTP Reflector hardware boosts the capacity to 120,000 NTP requests per second (1 Gigabit Ethernet speed). The reflector provides port-by-port packet limiting and DoS detection and alarming.

The following illustration shows an overview of the Microsemi Domain Time II.

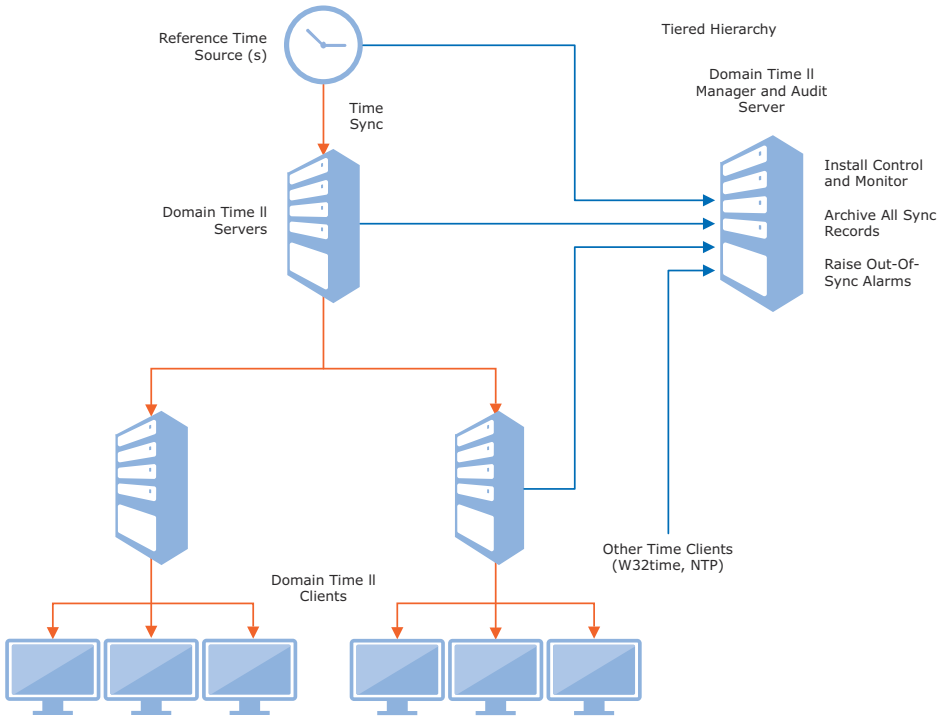

***Figure 2* • Domain Time II**

The server provides <15 nanosecond accuracy to UTC while tracking GPS with 24-hour holdover, depending on the server's oscillator. The oscillator types and their accuracies are as follows.

- Standard oscillator- 400 microseconds
- OCXO oscillator- 25 microseconds
- Rubidium oscillator- <1 microseconds (<3 microseconds at 3 days)

Holdover is the amount of error a clock accumulates if allowed to drift when its reference to GPS (or another external time source) is lost. This level of accuracy is more than sufficient to support the millisecond range use cases that Windows can support if W32Time is replaced.

Time Synchronization in Windows Networks is Critical

The fact is that for network and business operations, time synchronization is critical and the native Windows W32Time service does a poor job, by design. The good news is that most versions of Windows can handle millisecond-range (or better) use cases. Fortunately, implementing a trusted timekeeping solution that synchronizes system clocks and manages and monitors the time across the network is easy to deploy and very affordable. If network timing that is highly accurate, precise, scalable, reliable, manageable, and secure is at all important to your network and business operations, then you should seriously consider migrating from W32Time to a solution like Domain Time II that meets your requirements.