'Aging' Network Time Protocol (NTP) is Critical to Modern Architectures

Lori MacVittie, 2014-26-02

#infosec #DDoS #webperf NTP is rarely mentioned as a protocol of importance, but without it, high availability would be nigh unto impossible and performance enhancing services would be crippled

Krebs (on Security) writes about the recent NTP-based DDoS attacks in "The New Normal: 200-400 Gbps DDoS Attacks" and as is the case with most coverage of the attack, describes NTP as basically the Internet's time keeper. Which it is.

While generally described to broader audiences as "that server that keeps your computer up to date", it's less frequently (i.e. almost never) that the importance of NTP to modern infrastructure architectures is conveyed. In fact, it's so misunderstood that it was described as "aging" by one author with an almost derisive comment that it was "still employed by nearly every Internet-connected device."

Which is true - it is - but there's good reasons for that, reasons that go beyond simply being in sync with the Internet's internal clock.

Timestamps, Heartbeats and Microseconds

The way in which high availability architectures work is fairly rudimentary: keep tabs on the resource required to maintain availability and if it suddenly becomes unavailable, do something about it.

In cloud architectures this might be launch a new instance of the resources. In traditional multi-site architectures it might be redirect users to the secondary (failover) site. Within a network architecture it's definitely make the secondary (often the standby device in a redundant pair) the primary.

How the actual failover happens is dependent on what resource or network element has failed and how it's configured, but suffice to say it happens and that's what's important to this discussion. Of increasing importance is how quickly it's noticed that a device or resource has failed. The faster you recognize something has failed (or is about to fail, if you're lucky) the faster you can react and restore availability (or avoid it altogether, which is really the optimal goal).

Now, the question should be at this point, how does the system become aware that a device or resource has failed?

One of the primary methods is to use ICMP (for network-only elements) and TCP along with HTTP content verification checks for application elements. Regardless of the protocol used, they all carry with them an interesting little bit of information that is often ignored: a timestamp.

That timestamp is derived from the underlying operating system, which in turn is synchronized by - yes, you guessed it - NTP.

This timestamp is used for a variety of purposes in infrastructure elements. Availability is one of them, as most devices configured in a redundant pair or increasingly as part of a larger fabric need to be time synced as part of the configuration. Heartbeats and "status checks" between infrastructure elements uses this timestamp to determine whether or not other infrastructure elements are active. If the timestamp drifts or is markedly different, this can cause the infrastructure to believe something is wrong and take action - including failing over to its own secondary, initiating the provisioning of additional application instances, or simply marking application or infrastructure resources as "offline and unavailable." None of these are acceptable scenarios unless they are actually happening. NTP ensures that all systems - and thus timestamps - are accurate. When you're counting microseconds, that's a critical dependency.

But perhaps more important (and relevant to more people) is the impact of timestamps on performance-related services, such as caching. Cached content carries with it the notion of expiration, that the content should only be cached for X amount of time before it's refreshed from the origin. That time is determined by, you guessed it, timestamps.

Web performance monitoring systems, too, may be relying on those timestamps to document (in the office, permanent application record) the responsiveness of applications and services. More disturbing, perhaps, is the reality that a time-drift due to a non-synchronizing NTP service might be the difference between meeting an application performance SLA and, well, not meeting it.

Time synchronization is a key component to a well-oiled data center. More network and application elements depend upon this "aging" protocol than is apparent on the surface. And as we continue to adopt technologies that rely on the use of automation and orchestration to streamline processes - processes that often rely on timing - synchronization of time across the data center is going to continue to be an integral component.

So we shouldn't dismiss NTP because it's aging (so is IP, TCP, BGP, and DNS for that matter, but they're still the critical underpinnings of every network today) nor should we imply organizations should not continue to use it. Rather it's important to recognize and understand ***why*** organizations still rely on NTP and what capabilities it is enabling under its admittedly simple purpose, as well as why we will continue to rely on it until some other means of measuring and synchronizing time across systems, devices, and networks is discovered.

F5 Networks, Inc. | 401 Elliot Avenue West, Seattle, WA 98119 | 888-882-4447 | f5.com

F5 Networks, Inc.	F5 Networks	F5 Networks Ltd.	F5 Networks
Corporate Headquarters	Asia-Pacific	Europe/Middle-East/Africa	Japan K.K.
info@f5.com	apacinfo@f5.com	emeainfo@f5.com	f5j-info@f5.com