MARVELL

**White Paper**

# Zen and the Art of Network Timestamping

Tal Mizrahi

Switching Architecture Marvell

December 2021

MARVELL


## ABSTRACT

Timestamps play a key role in modern communication networks, enabling a variety of important tasks to be executed. The objective of this white paper is to serve as a comprehensive resource on the subject of timestamping. It will describe the diverse array of applications that utilize timestamps, outline some of the major challenges when it comes to implementing accurate timestamping in network devices and then how a new approach introduced by Marvell will enable more effective network timestamping functions to be realized.


## 1. Introduction

Network devices, namely switches and routers, are used for forwarding data packets from their source to their destination - or at least that is what they are meant to do. In practice, these devices tend to do a lot more than that. They can be involved in Quality of Service (QoS) enforcement, filtering, load balancing, fault detection, performance measurement, event logging and various other activities. Increasingly these functions require accurate timing references and, as a result, the constituent network devices must be able to support timestamps.

Timestamps are employed in network devices for various purposes - measuring network delays and performance monitoring being among the most common of these. In addition, timestamps are attached to packets for sampling and analysis purposes. Timestamps are also used in logs and reports to record the time of occurrence for events. In the last few years network timestamping has seen greater uptake. This has been driven by the emergence of more accurate time synchronization technology, along with the introduction of the IEEE 1588 [1] synchronization protocol.

This white paper briefly discusses some of the major applications that require network timestamping. It will look at the challenges of implementing network timestamping and what needs to be done to overcome these. Then it will introduce Marvell's generic and flexible approach to network timestamping and the benefits that this will have as timestamping continues to develop as an art form.

## 2. Network Timestamps Explained

In a nutshell, a timestamp is a snapshot of the wall-clock time. A network device typically associates a timestamp with a specific 'packet' or a specific 'event'.

A packet timestamp is used for specifying the instant at which a packet was forwarded through a network device. Specifically, we distinguish between two distinct terms:

**a. The ingress timestamp** – This specifies the instant at which the first bit of the packet is received by the device.
**b. The egress timestamp** – This specifies the instant at which the first bit of the packet subsequently transmitted out of the device.

By measuring both the ingress timestamp and egress timestamp, a device can compute the residence time of a packet, reflecting how long a packet has spent in the current network device.

Frequently timestamps are associated with events, rather than packets. For example, it is often useful to bind a timestamp to events such as system initialization or port failure. Timestamps can have merit in this context when these events are analyzed by the network management system.

## *What Does a Timestamp Look Like?*

Timestamps are used in network protocols in two main forms:

a. **Text-based timestamps** [2], [3] – These are a human-friendly representation of the time-of-day. For example:

   2017-07-26T00:00:00Z

   Text-based timestamps are widely used in MIBs, YANG models, and in various JSON or XML based information models.

b. **Packet timestamps** [4] – These are generally represented in a more compact way than text-based timestamps. Furthermore, they have a fixed length, as they are intended to be used in packet header formats. Two examples of common packet timestamp formats are the Network Time Protocol (NTP) timestamp [5] and the Precision Time Protocol (PTP) timestamp format [1].

## *How are Timestamps Used?*

A timestamp is typically used by a network device in one of two ways:

a. **In-band timestamping** – It is often necessary for network devices to incorporate timestamps in packets, as some network protocols require timestamps to be inserted into en-route packets.

b. **Timestamp logging** – Timestamps can be used for monitoring events or in logs. In such cases a network device will store the timestamp that corresponds to a specific packet or event, along with other information that is relevant to that packet or event.

## *The Need for In-band Timestamping*

In-band timestamps are proving themselves to be of benefit in various network applications and network protocols. For example, the Transmission Control Protocol (TCP) congestion control relies heavily on the ability to measure the round-trip-time [6] via timestamps. Numerous performance monitoring protocols [7], [8] use timestamped packets for measuring the network delay.

Another interesting application of timestamps is in-band telemetry [9]. Here each network device along the path incorporates timestamps (and potentially other information) in the headers of data

plane packets, allowing fine-grained measurement and congestion detection. These approaches are known as In-band Network Telemetry (INT) [10] and In-situ OAM (IOAM) [11].

Incorporating timestamps in some or all of the data plane packets [12], [13] is a powerful tool that can be used not only for telemetry, but also for taking smart decisions that concern network operation. This will be described in detail within the next section.

# 3. Taking Smart Decisions Based on Timestamps

Network policies that depend on the time-of-day have been widely used for many years. For example, time-of-day routing changes the network paths as a function of the time; peak hours require more network resources, while in off-peak hours the network paths can be arranged in a way that utilizes less resources. In-band timestamps enable even smarter and more powerful processing than time-of-day routing.

## *Consistent Network Updates*

Consider a scenario in which it is necessary to update the forwarding policy in the network - from Policy A to Policy B. The policy affects the processing of multiple devices in the network, and it is required that the migration from A to B to occurs consistently, with every packet being forwarded either according to Policy A, or according to Policy B, but not according to a mixture of the two policies [14].

Even if all the network devices have been updated to start using Policy B at the exact same time instant, packets that are already en-route during the update may be forwarded inconsistently.

The in-band timestamp can be used for consistently applying the new policy, as the network devices can be instructed to apply Policy A if the timestamp is $<T_0$, and to apply Policy B if the timestamp is $\geq T_0$. If every packet includes an in-band timestamp, this guarantees that a uniform and consistent behavior is applied across the whole network without any policy irregularities being witnessed.

## *Alternate Marking*

Alternate marking [15] is a method of measuring loss and delay between two Measurement Points (MPs) via a single bit in the header of every packet. Basically, the header of each data packet includes a binary color bit, either ‘0’ or ‘1’. The color bit divides the traffic into consecutive blocks of packets (Figure 1), allowing the two MPs to measure each block separately. The alternating color allows very accurate measurement of the loss and delay between the two MPs.

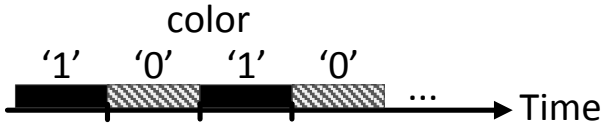


**Figure 1: The Alternate Marking Method**

The color is toggled periodically, so that each color is used for a fixed time interval. Hence, the color bit can be viewed as a one-bit timestamp that wraps around periodically. Moreover, if the data packets already carry an in-band timestamp, then it is possible to use one of the timestamp bits as the color bit. For example, if the timestamp is measured in seconds, choosing the least significant bit of the timestamp results in a color bit that is toggled with a one-second period.

### *How to Apply Timestamp-Based Decisions*

A practical question that should be asked at this point is: How can a smart decision based on the timestamp be applied within a hardware-based network device? Clearly there may be various different implementations that allow accurate decision making based on timestamps.

TimeFlip [16] is a method that enables timestamp-based decisions to be taken in hardware network devices in an accurate and efficient way. TimeFlips are implemented using Ternary Content Addressable Memory (TCAM) resources. A TCAM is a common building block in network devices. It is a memory for fast searches, in which each bit may have three possible values, '0', '1', or 'don't care'. The don't care value can be used for representing ranges of values. A TimeFlip is a TCAM entry that includes the timestamp field. By including the timestamp field in the TCAM search key it is possible to define time ranges, as illustrated in Figure 2. Here a timestamp-based TCAM entry, with don't care values in all the bits except the least significant bit of the seconds, yields a period timestamp range with a one-second period.

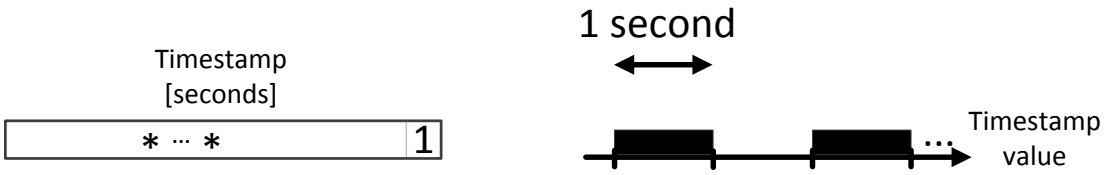


**Figure 2: Schematic Showing a TimeFlip Example**

TimeFlip is a simple and accurate method of applying policies and decisions that are confined to a specific range of times. Notably, TimeFlip relies on the inherent properties of TCAMs, and is therefore simple to implement.

## 4. Perfecting the Art of Timestamping

The element that makes timestamping a bit tricky is that timestamps need to be accurate and precise. Figure 3 illustrates the difference between these two terms. Accuracy is defined [17] as "the quality of being near to the true value", while precision is defined as "the quality of being reproducible in amount or performance."

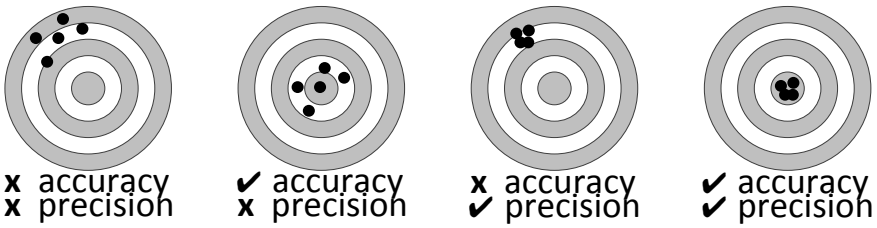


**Figure 3: The Difference Between Accuracy and Precision**

## *Accurate and Precise Timestamping*

**Accurate timestamps -** In order for timestamps to be accurate, they must be taken with respect to an accurate clock. In network devices accurate clocks are often implemented in hardware, and are synchronized to an accurate time source using a synchronization protocol, such as PTP. A hardware-based clock that is synchronized using PTP can reach an accuracy of the order of 1 microsecond or less in typical PTP-enabled networks.

**Precise timestamps** - A precise timestamp implies that the wall-clock time value must be measured with a high degree of exactitude at the point in time when the corresponding event occurs. Thus, an ingress timestamp requires the time to be captured at the precise instant that the first bit of the packet is received. Similarly, an egress timestamp should correspond to the instant at which the first bit of the packet is transmitted. Thus, the arrival or departure times must be measured as close as possible to the device's physical interface.

Any element that causes unknown or non-deterministic delay will compromise the precision of the timestamp. Typical sources of uncertainty include FIFOs and clock domain crossings, as illustrated in Figure 4.

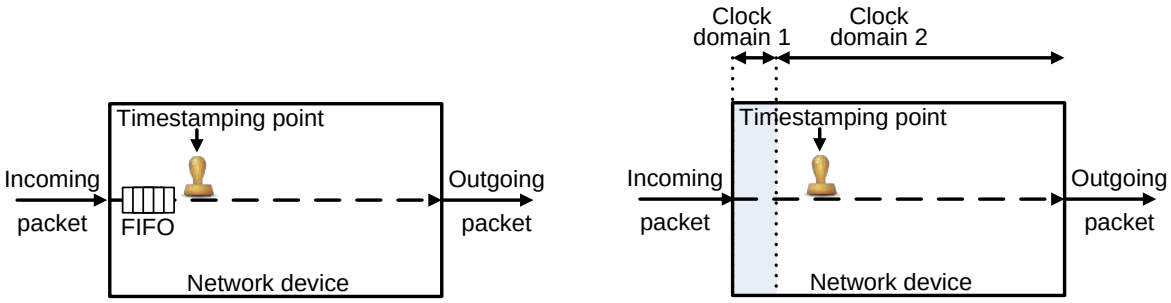


**Figure 4: Potential Causes for Imprecision in the Timestamping Process**

## *Timestamping and Encryption*

So, what happens if a timestamped packet needs to be sent through an encrypted connection? Timestamped encrypted traffic introduces a dilemma. On one hand, the expectation is that the entire packet should be encrypted, including the timestamp, and thus the encryption function must be performed after timestamping. On the other hand, if the timestamp is to represent the

The way to resolve this problem is to implement an encryption module that has deterministic delay. If the latency experienced is a function of the packet length, and can be known before the packet is encrypted, then the time can be measured before it enters the encryption engine. The corresponding in-band timestamp should thus represent a future value that takes the constant delay of the encryption engine into account.

## *Timestamps and Checksums*

Some of the most widely used network protocols, including TCP and User Datagram Protocol (UDP), rely on a packet checksum, which is intended for error detection. Pushing a timestamp into an en-route packet that is protected by a checksum is challenging, as the checksum must be updated to reflect the new packet, after having been timestamped.

UDP packets sent over IPv4 network infrastructure can include a zero checksum value. In such cases the receiver of the packet simply does not verify the packet checksum. Therefore, network devices that perform en-route timestamping often assign zero to the checksum in order to avoid the nuisance of updating the checksum value.

However, the zero checksum approach is only applicable to UDP over IPv4. Specifically, in UDP over IPv6 the checksum is mandatory. The good news is that the UDP checksum can be updated incrementally [18], i.e. by considering only the packet fields that were modified. The bad news is that hardware devices often perform serial processing of the packet, as illustrated in Figure 5. Consequently, if the timestamp field is updated on step 2 (as denoted in the figure), it may prove to be too late to update the value of the checksum.

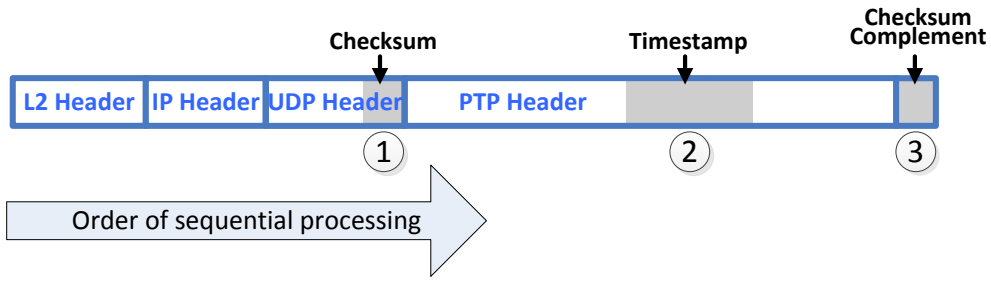


**Figure 5: Checksum Updates in Hardware Devices that Perform Sequential Processing**

An elegant way to resolve this issue has been introduced in the IEEE 1588 protocol [1]. The idea is that a Checksum Complement field at the end of the packet is reserved for the checksum computation. Referring to Figure 5, firstly the Checksum field is read (step 1), but not modified. After the timestamp field has been updated (step 2), the Checksum Complement value is updated to a new value (step 3), such that the existing Checksum field (in the UDP header) is correct for the updated packet. The Checksum Complement approach has also been adopted by other network protocols [11], [19], [20].

## 5. Marvell's Timestamping Approach

As discussed on the previous sections, network timestamping often requires high levels of precision, and is being employed within a diverse range of applications. In order to address this diversity while maintaining the accuracy needed, Marvell's approach to network timestamping is one that is inherently flexible. As a result it can not only address a broad range of potential application demands, it also recognizes the fact that the number of protocols that now use timestamps is increasing all the time.

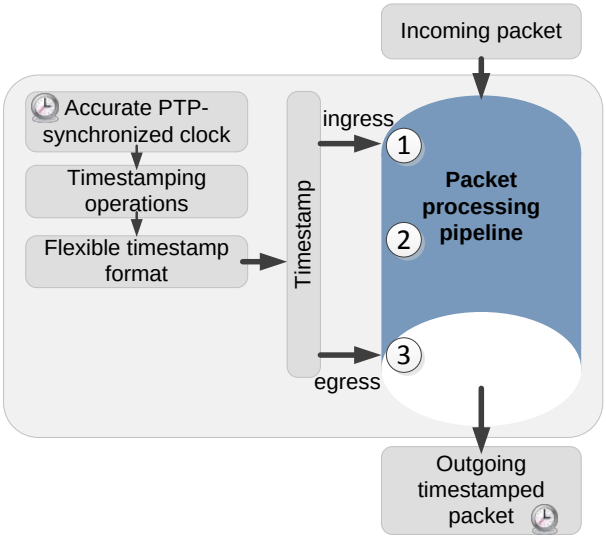


**Figure 6: Description of Marvell's Network Timestamping Approach**

Conventional thinking says that a timestamping mechanism must be very lightweight and efficient, as it is implemented by hardware logic that must be as close to the wire as possible. Consequently, this paradigm implies that timestamping logic is typically fixed and customized to a specific protocol and timestamp format.

Figure 6 gives details of how Marvell, using its innovative Prestera Ethernet switch technology is providing the industry with a more streamlined but still effective and versatile timestamping solution. Each packet has an ingress timestamp attached to it (denoted by '1' in the figure) and this timestamp value can be used in the packet processing ('2' in the figure). If necessary, the packet's egress timestamp is measured ('3' in the figure).

Optionally, the timestamp can be inserted into the packet. Both the ingress and egress timestamping operations are performed to a high level of accuracy since the timestamp is sampled as close as possible to the device's interface.

## Accurate PTP-synchronized Clock

Timestamping is performed using the packet processor's internal hardware timer. This timer can be synchronized to an accurate network grandmaster using PTP, allowing highly accurate network-wide timestamping.

## Ingress Timestamping Across the Board

Key to the Marvell solution is its ability to timestamp every incoming packet at full-wire-speed without operational performance being impinged upon. As illustrated in step 1 of Figure 6, each packet has an internal ingress timestamp attached to it. This internal timestamping is performed in full-wire-speed, without any reduction in the device's bandwidth, and without increasing the size of the packet. The internal timestamp can be used for various synchronization and delay measurement protocols [1], [7], [8], [11]. Notably, the internal timestamp can also be used for smart timestamp-based decisions, as described in Section 3.

## Flexible Encapsulation-Agnostic Parsing

A key feature of the Marvell solution is flexible encapsulation-agnostic parsing. Through this the device is able to identify and subsequently process timestamp-related packets. This is done by either:

a. **Common Encapsulation Parsing** – Where the device's header parser identifies the common packet formats and encapsulations that require timestamping. For instance, PTP packets encapsulated in one of the standard transport types (defined by the IEEE 1588 standard), are detected and parsed by the device's header parser.
b. **Flexible Encapsulation Parsing** – Where parsing is executed for non-conventional encapsulations using a TCAM lookup. The TCAM allows full flexibility in detecting the packet header and consequently performing the required timestamping-related processing.

Because of its encapsulation agnostic parsing, the Marvell approach is future proof, as it allows timestamping to be carried out not only over any current forms of encapsulation, but emerging ones too.

## Diverse Timestamping Operations

Timestamps are utilized by various different network protocols and applications. However, using a small set of timestamping operations it is possible to implement the diverse applications described in the previous sections.

The set of timestamping primitives are:

- Push ingress time
- Push egress time
- Push residence time
- Log/report ingress time
- Log/report egress time
- Take decision based on ingress time (TimeFlip)

Delay measurement protocols typically require ingress/egress timestamps to be pushed into the protocol packets. The residence time of the switch or router is useful in network telemetry protocols and in time synchronization protocols (like PTP). Network flow monitoring protocols such as IPFIX [21] often require packet timestamps to be logged and reported.

### *Smart Packet Processing using TimeFlips*

Since a timestamp is internally attached to every packet, the device can take time-based decisions using TimeFlip. Moreover, TimeFlip can be applied either on an in-band timestamp that resides in the packet header, if one is present, or on the internal timestamp, without any overhead on the packet header.

### *Flexible Timestamp Formats*

Even though the set of primitives above is small, different applications and protocols often use different timestamp formats. Therefore, Marvell's more generic timestamping approach enables the use of multiple timestamp formats, such as the PTP timestamp format [1], the NTP timestamp format [2] and others. In order to allow timestamping over a UDP transport, Marvell's timestamping mechanism also permits the UDP Checksum Complement to be updated, thereby making on-the-fly checksum correction possible.

## 6. Conclusion

Timestamps are a fundamental tool for carrying out measurement in modern day communication networks. Accurate timestamping is a delicate art that requires careful and flexible implementations. Marvell's approach to accurate in-band timestamping provides the flexibility needed to avoid having to reinvent the mechanism for each specific use case. It supports various existing network protocols that use in-band timestamping, as well as having provision to accommodate new protocols when they are introduced.

As networks continuously evolve towards software-defined and automated environments, the ability to use time and timestamping is establishing itself as an essential feature. Moreover, as network telemetry becomes increasingly important in high-speed communication infrastructure, the value of timestamping is destined to grow still further.

## About the Author

**Tal Mizrahi, PhD**
*Feature Definition Architect*

Tal Mizrahi is a feature definition architect at Marvell. With over 15 years of experience in networking, network security and ASIC design, Tal has served in various positions in the industry, including system engineer, team leader and, for the past 10 years, an architect for Marvell's networking product line. Tal received his BSc., MSc. and Ph.D. in Electrical Engineering from the Technion, Israel Institute of Technology. Tal is an author of over 40 published patents, and over 25 academic publications. He is also an active participant in the Internet Engineering Task Force (IETF).

# References


[1] IEEE TC 9, "1588 IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems Version 2", IEEE, 2008.

[2] Klyne, G. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, DOI 10.17487/RFC3339, July 2002, <http://www.rfc-editor.org/info/rfc3339>.

[3] "Data elements and interchange formats – Information interchange -- Representation of dates and times", ISO 8601:1988(E), International Organization for Standardization, June, 1988.

[4] Mizrahi, T., Fabini, J. and A. Morton, "Guidelines for Defining Packet Timestamps", draft-ietf-ntp-packet-timestamps (work in progress), 2017, <https://tools.ietf.org/html/draft-ietf-ntp-packet-timestamps>.

[5] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <http://www.rfc-editor.org/info/rfc5905>.

[6] Jacobson, V., Braden, R., and D. Borman, "TCP Extensions for High Performance", RFC 1323, DOI 10.17487/RFC1323, May 1992, <http://www.rfc-editor.org/info/rfc1323>.

[7] ITU-T, "OAM functions and mechanisms for Ethernet based Networks", ITU-T Recommendation G.8013/Y.1731, August 2015.

[8] Frost, D. and S. Bryant, "Packet Loss and Delay Measurement for MPLS Networks", RFC 6374, DOI 10.17487/RFC6374, September 2011, <http://www.rfc-editor.org/info/rfc6374>.

[9] C. Kim, A. Sivaraman, N. Katta, A. Bas, A. Dixit, and L. J. Wobker, "In-band network telemetry via programmable dataplanes", ACM SIGCOMM Symposium on SDN Research (SOSR), 2015.

[10] C. Kim et al., "In-band network telemetry (INT)", P4 consortium, 2015.

[11] Brockners, F., Bhandari, S., Pignataro, C., Gredler, H., Leddy, J., Youell, S., Mizrahi, T., Mozes, D., Lapukhov, P., Chang, R., and D. Bernier, "Data Fields for In-situ OAM", draft-ietf-ippm-ioam-data (work in progress), 2017, <https://tools.ietf.org/html/draft-ietf-ippm-ioam-data>.

[12] Mizrahi, T., Yerushalmi, I., Melman, D., Browne, R., "Network Service Header (NSH) Context Header Allocation: Timestamp", draft-mymb-sfc-nsh-allocation-timestamp (work in progress), 2017, <https://tools.ietf.org/html/draft-mymb-sfc-nsh-allocation-timestamp>.

[13] Mizrahi, T. and Y. Moses, "The Case for Data Plane Timestamping in SDN", IEEE INFOCOM Workshop on Software-Driven Flexible and Agile Networking (SWFAN), 2016.

[14] Reitblatt, M., Foster, N., Rexford, J., Schlesinger, C. and D. Walker, "Abstractions for network update", ACM SIGCOMM, 2012.

[15] Fioccola, G., Capello, A., Cociglio, M., Castaldelli, L., Chen, M., Zheng, L., Mirsky, G., and T. Mizrahi, "Alternate Marking method for passive and hybrid performance monitoring", RFC 8321, 2018, <http://www.rfc-editor.org/info/rfc8321>.

[16] Mizrahi, T., Rottenstreich, O. and Y. Moses, "TimeFlip: Scheduling Network Updates with Timestamp-based TCAM Ranges", IEEE INFOCOM, 2015.

[17] "The Free Dictionary", <http://www.thefreedictionary.com>.

[18] Rijsinghani, A., Ed., "Computation of the Internet Checksum via Incremental Update", RFC 1624, DOI 10.17487/RFC1624, May 1994, <http://www.rfc-editor.org/info/rfc1624>.

[19] Mizrahi, T., "UDP Checksum Complement in the Network Time Protocol (NTP)", RFC 7821, DOI 10.17487/RFC7821, March 2016, <http://www.rfc-editor.org/info/rfc7821>.

[20] Mizrahi, T., "UDP Checksum Complement in the One-Way Active Measurement Protocol (OWAMP) and Two-Way Active Measurement Protocol (TWAMP)", RFC 7820, DOI 10.17487/RFC7820, March 2016, <http://www.rfc-editor.org/info/rfc7820>.

[21] Quittek, J., Zseby, T., Claise, B., and S. Zander, "Requirements for IP Flow Information Export (IPFIX)", RFC 3917, DOI 10.17487/RFC3917, October 2004, <http://www.rfc-editor.org/info/rfc3917>.

**Marvell Semiconductor, Inc.**
5488 Marvell Lane
Santa Clara, CA 95054, USA

Tel: 1.408.222.2500
www.marvell.com