Time-based Updates in Software Defined Networks

Tal Mizrahi
Technion — Israel Institute of Technology
dew@tx.technion.ac.il

Yoram Moses*
Technion — Israel Institute of Technology
moses@ee.technion.ac.il

ABSTRACT

Network configuration updates are a routine necessity, and must be performed in a way that minimizes transient effects caused by intermediate states of the network. This challenge is especially critical in Software Defined Networks, where the control plane is managed by a logically centralized controller, and configuration updates occur frequently. In this paper we discuss the tradeoff between maintaining consistency during configuration updates and the update performance. We introduce an approach that uses time to coordinate network configuration and reconfiguration. We show a simple time-based configuration approach called TIMECONF and show that this approach offers significant advantages over existing update approaches at the cost of a brief inconsistency. We also show that time can be used as a tool for simplifying existing update approaches without compromising consistency.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Dist. Systems.

Keywords

SDN, clock synchronization, configuration, management, time.

1. INTRODUCTION

The Configuration Update Problem. Software Defined Networks (SDN) define a centralized architecture, where the control plane is managed by a controller. This centralized approach introduces various challenges in terms of consistency and performance. Traditional network management systems are in charge of initializing the network, monitoring it, and allowing the operator to apply occasional changes when needed. The SDN approach, on the other hand, requires the controller to routinely perform frequent configuration updates in the network. The controller must take care to minimize network anomalies during update procedures, such as packet drops or misroutes caused by temporary inconsistencies. Updates must also be planned with performance in mind; update procedures must scale with the size of the network, and thus cannot be too complex. The controller must be able to respond quickly to network events such as failures, topology changes or arrival of new flows. In the face of rapid configuration changes, the update mechanism must allow a high update rate.

Consistency vs. Performance. The CAP theorem [1] presents a tradeoff between Consistency, Availability and tolerance to network Partitioning (CAP for short) in distributed systems. Paraphrasing the CAP paradigm, we present a tradeoff between consistency and performance. Per-packet consistency [2] guarantees that every packet sent through the network is processed according to a single configuration version, either the previous or the current one. In this paper we show that the update performance can be improved when relaxing the consistency requirement.

Using Time in Distributed Systems. While the notion of using time to trigger events in distributed systems is certainly not new (e.g., [3]), time-based triggers were never considered practical in the context of network management due to the inaccuracy of network time synchronization. Neither OpenFlow [4] nor common management and configuration protocols, such as SNMP [5] and NETCONF [6], use accurate time for scheduling or coordinating configuration updates. However, clock synchronization in packet networks has evolved over the last few years. The Precision Time Protocol (PTP) [7] can synchronize clocks in a network to a very high degree of accuracy, typically on the order of microseconds. Thus, accurate time appears to be an accessible and useful tool for coordinating configuration changes.

TIMECONF. In this paper we introduce TIMECONF, a simple time-based configuration approach. We show that TIMECONF has significant advantages over existing approaches; it is simple, it allows fast and rapid updates and does not require modification of traversing packets. We also show that a time-based update preserves consistency better than other approaches in configuration changes that affect physical layer attributes.

We present a tradeoff between update consistency and performance, and show that at the cost of a brief inconsistency TIMECONF allows better update performance than other approaches.

We also show that time can be used as a tool to simplify existing consistent update approaches. We present a consistent variant of TIMECONF and discuss its performance cost.

2. ANALYZING TIMECONF

In TIMECONF the controller enforces coordinated updates by incorporating a scheduled execution time, T, in every configuration message. Thus, every update procedure starts with an offline preprocessing stage, where the controller computes the update time T, and distributes the configuration messages. Consequently, every switch executes the configuration update at the scheduled time, T, and thus all updates are performed during the period $(T-\delta, T+\delta)$, where δ denotes the clock synchronization accuracy in the system.

*The Israel Pollak academic chair at Technion; this work was supported in part by the ISF grant 1520/11.

HotSDN'13, August 16, 2013, Hong Kong, China.
ACM 978-1-4503-2178-5/13/08.

Packets that are en-route during the configuration update may be subject to inconsistent processing. Note that this is valid even when the clocks are globally synchronized. Assume the maximal packet delay in the network is bounded by D. Interestingly, consistency is not guaranteed during the period $(T-\delta-D, T+\delta)$, but is guaranteed before and after this period.

Existing Approaches for Consistent Updates. A common approach for avoiding inconsistencies in configuration updates is the ***sequential approach***, which uses a sequence of updates (e.g., [8, 9]), whereby the order of execution guarantees that no anomalies are caused at intermediate states of the procedure. Another recently introduced approach is the ***version approach*** [2], in which all packets are stamped with a configuration version tag, indicating whether they should be processed by the new configuration or by the old one. This allows each packet to be processed by the same configuration at all switches along its path. In this section we compare TIMECONF to these two existing approaches.

Per-Packet Consistency. While the version approach guarantees per-packet consistency, the sequential approach does not always guarantee consistency; for example, if the network topology is a layered directed acyclic graph, configurations can be applied sequentially to each layer so as to ensure a smooth transition. TIMECONF yields a brief period of inconsistency, but can be used regardless of the network toplogy and the nature of the update.

Physical Layer Consistency. Some configuration attributes, such as switch resource allocation or port transmission rate, are not assigned on a per-packet basis, and thus in these cases the per-packet consistency metric cannot be used. We refer to such updates as *physical layer updates*. In these cases it is natural to use a time-based approach.

Completion Time. Intuitively, the completion time of an update is a measure of how long it takes the system to migrate from the old configuration to the new one. Both the version approach and the sequential approach require a multi-step procedure, whereby each step involves a request-acknowledge handshake. In TIMECONF the controller computes the update time T, distributes the configuration updates to the switches in a single step, and then the switches perform the update based on the scheduled time, T. Since TIMECONF does not require multiple message exchanges or handshakes, its completion time can be significantly shorter.

Update Rate and Memory Consumption. While in the sequential approach updates are executed by the switch immediately upon reception, in the version approach the old and the new configurations have to be active concurrently, and are thus stored in expensive memories, such as Ternary Content-Addressable Memories (TCAM). This implies that switches have to maintain a large number of costly memory entries to allow consistent updates. An alternative approach is to maintain a small number of excess entries for configuration updates, limiting the number of concurrent updates, and hence the update rate.

In TIMECONF only one configuration is used at any given time. When the switch receives an update message it stores the new configuration for a short transition period until the scheduled update time, but this information can be stored in a relatively cheap DRAM, rather than in an expensive TCAM.

Transparency. In the version approach packets are tagged with the version number, distinguishing between packets handled by the old configuration and packets handled by the new one. TIMECONF is transparent to the network, as it does not require any modifications to traversing packets.

Simplicity of the Update Procedure. TIMECONF is simple and stateless, while the two reference approaches require the controller to apply stateful multi-step procedures.

3. CONSISTENT TIME-BASED UPDATES

Consistent TIMECONF with Selective Store and Forward. A slightly tweaked variant of TIMECONF can be adopted for "sensitive" traffic. Such traffic can be stored at the switches during the short transition period, and then forwarded when the configuration update is completed. This approach requires switches to store a relatively small number of packets during the short transition period, and guarantees per-packet consistency for this traffic.

Consistent TIMECONF with Versioning. Time can be used as a tool for simplifying the version approach; every update message includes a schedule, defining when switches start using the new version and when they stop supporting the old one. The schedule can be carefully determined in a way that guarantees consistency. Thus, the controller can distribute the configuration update messages in a single step, significantly simplifying the update procedure.

Consistent TIMECONF with Sequential Updates. Extending the sequential approach, we define time-based sequential updates, where the controller defines a set of update times, $T_1, T_2, \ldots, T_n$ that determines the order in which updates are performed. This approach, much like the sequential approach, is only applicable in cases where consistency can be guaranteed by the order of updates.

A Hybrid Approach. In real-life networks a hybrid approach can be adopted, where critical configuration changes that require quick response and scalability are applied using TIMECONF, whereas configuration changes that require packet consistency use other configuration approaches.

4. CONCLUSION

Time can be a powerful tool in network configuration and reconfiguration; the controller can compute the optimal update time for each switch based on the nature of the attributes being updated. Time can be used to guarantee a specific order of update events, or to invoke a multi-step update procedure that guarantees consistency by maintaining both the old and the new configuration during the update procedure. In its most light-weight form, time-based configuration can be used to invoke simultaneous updates, at the cost of a short period of inconsistency.

It is only natural to consider time-based reconfiguration as a future extension to the OpenFlow protocol, as well as to standard network management protocols such as SNMP and NETCONF.

5. REFERENCES

[1] E. A. Brewer, "Towards robust distributed systems," in *Proceedings of PODC 2000*, vol. 19, pp. 7–10, 2000.

[2] M. Reitblatt, et al., "Abstractions for network update," in *Proceedings of ACM SIGCOMM*, pp. 323–334, 2012.

[3] L. Lamport, "Using time instead of timeout for fault-tolerant distributed systems.," *ACM Trans. Program. Lang. Syst.*, vol. 6, pp. 254–280, Apr. 1984.

[4] Open Networking Foundation, "Openflow switch specification," *Version 1.3.0*, June 2012.

[5] J. Case, et al., "A simple network management protocol (SNMP)," RFC 1157, IETF, 1990.

[6] R. Enns, et al., "Network configuration protocol (NETCONF)," RFC 6241, IETF, 2011.

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

[8] P. Francois, et al., "Avoiding transient loops during the convergence of link-state routing protocols," *Trans. on Networking*, vol. 15, no. 6, pp. 1280–1292, 2007.

[9] S. Bryant, et al., "IP Fast Reroute using tunnels," draft-bryant-ipfrr-tunnels, work in progress, IETF, 2004.