October 01, 2018

Management of IEEE 802.1Qci Security Policies for Time Sensitive Networks (TSN)

Robert Barton

Maik Seewald

Jerome Henry

Management of IEEE 802.1Qci Security Policies for Time Sensitive Networks (TSN)

AUTHORS:
Robert Barton
Maik Seewald
Jerome Henry

ABSTRACT

Techniques are described herein to provide a centralized policy management mechanism that enables: the consumption of traffic intent from the well-known CNC; translation into new rule sets dynamically applied along the path; and the dynamic monitoring of the CNC rule sets for dynamic adaptation of rules as the traffic intent changes.

DETAILED DESCRIPTION

The IEEE 802.1 standard series defines Time Sensitive Networking (TSN) for communications of latency sensitive applications. These applications are used in manufacturing, utilities, and many other industries. One of the TSN sub-standards is IEEE 802.1Qci, which defines policing and filtering on ingress ports to protect time-sensitive flows. This is an important security addition to TSN because it ensures protection against excessive bandwidth usage, burst sizes, and incorrectly configured or malicious endpoints. 802.1Qci can be further used as a mechanism to isolate faults to specific regions of the network, thus limiting the impact to other parts of the network.

Although 802.1Qci is a published standard that defines the protection of TSN flows in a shared Ethernet network, little progress has been made to connect the standard with existing industrial security systems and architectures. The 802.1Qci approach is network-element-centric, and focused on buffer efficiency. Furthermore, very little has been explored about how 802.1Qci policies could be deployed on network devices and merged with existing industrial security policies on those devices.

The 802.1Qci standard is relatively new and does not define how specific policies are created and deployed. In addition, 802.1Qci limits its scope to defining how a network element (e.g., switch) reads incoming traffic, compares the entering flow to traffic importance labels, and applies possible filtering to avoid unidentified or low importance traffic to delay more important traffic. However, the 802.1Qci standard does not define how these filtering rules appear or are dynamically updated at the scale of the network.

Techniques disclosed herein fill that gap, with methods to deploy intent-based, per-flow 802.1Qci firewall rules on TSN bridges. These techniques integrate the security of 802.1Qci TSN flows with a generalized security architecture in industrial networks, effectively allowing industrial switches that implement 802.1Qci to also *de facto* become industrial firewalls.

TSN networks following a centralized configuration model employ a Central User Configuration (CUC) that defines the overall business intent of each TSN application. The CUC, in turn, communicates with a Central Network Controller (CNC) that acts as a proxy and network configuration component for the TSN bridges and their endpoint connections. Through a restful API, the CNC learns all TSN flow information from the CUC, including sending entity or target entity MAC addresses and the application latency requirements of the flow. With this information, the CNC is able to discover the intended network path for all present flows, and program the appropriate flow schedule into all TSN bridges.

According to 802.1Qci, the CNC does not create or configure the security policies or firewall rules. Instead it only describes bridge configurations that allow shaper mechanism and schedules. This is because security on TSN bridges/switches encompasses more than just the TSN flows.

To configure and deploy security policy rules based on 802.1Qci, a centralized view of security is proposed. Examples of a centralized policy server may include identity servers, an SCL configuration system (for utility), or an OPC configuration instance (in manufacturing environments). With reference to Figure 1, the proposed method functions as follows:

1. To centralize the security policy, a connection is made from the Policy Server to the CNC. The identification and profiles (business intent) of TSN endpoints are imported as managed objects. Further, the TSN network communication paths defined by the CNC are imported as abstracted security zones and conduits.
2. The security policy server owns the global security policies for the whole network. These policies are typically translated into specific configurations,

which are then provisioned on the networking devices. These existing policies are required to merge with the dynamic 802.1Qci TSN security policies. Both are translated into switch-specific configurations and are deployed accordingly. A mechanism is built to translate the business intent of the CUC into security network configuration.

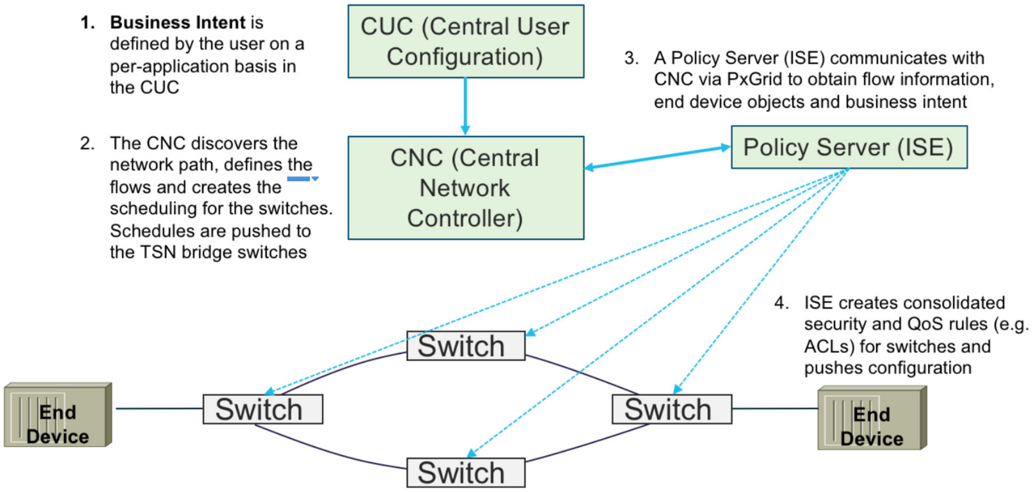
1. Business Intent is defined by the user on a per-application basis in the CUC
CUC (Central User Configuration)
2. The CNC discovers the network path, defines the flows and creates the scheduling for the switches. Schedules are pushed to the TSN bridge switches
CNC (Central Network Controller)
3. A Policy Server (ISE) communicates with CNC via PxGrid to obtain flow information, end device objects and business intent
Policy Server (ISE)
4. ISE creates consolidated security and QoS rules (e.g. ACLs) for switches and pushes configuration
End Device
Switch
Switch
Switch
Switch
End Device

Figure 1

3. The switches are expected to have existing security rules (e.g., DAI, MAC filtering, ingress ACL rules, *etc*.), based on the industrial network security policy. The Policy Server dynamically adds or removes the dynamic 802.1Qci rules from the switches as the CNC creates a path with specific bandwidth requirements through the network. In addition, the Policy Server correlates and harmonizes the 802.1 Qci filtering and policing rules with the existing rules in order to create a consistent security mechanism. As a result, the Policy Server acts as a dynamic firewall rule generator on the TSN switches to ensure that the TSN flows are protected and that any misconfiguration does not expose the TSN endpoints to random parts of the network.
4. Furthermore, as new TSN flows are added, the QoS configuration on the switches would need modification. The CNC is responsible for per-flow shapers, but the remainder of the QoS policy/configuration is adapted as more TSN flows are present. TSN flows always supersede other flows, thus reducing the available bandwidth on the port. The Policy Server also adjusts the QoS

configuration with respect to queue and buffer sizes, in relation to the number of TSN flows that are present. This happens automatically (based on intent and context) and immediately. No manual configuration is needed.

5. The Policy Server either configures the switches directly, or communicates with an orchestration server, which provisions the full security configuration to the switches and TSN bridges.

A policy engine/server holds policies that associate objects to authentication or authorization profiles. In addition, a policy server according to the present proposal consumes the configuration of the automation system from a well-defined component, the TSN CNC. This component does not express security policies, but rather the traffic intent between the latency-sensitive endpoints.

The policy server combines and correlates the information (intent and context) retrieved from 802.1Qci traffic intent files, to create a consistent and comprehensive policy along the intended path. Furthermore, the policy server monitors the files present in these CNC components to dynamically absorb and parse new file versions and to adapt the corresponding policy to any change in the configuration of the automation system, such as changed stream definitions or intent patterns. In other words, any new flows or changes in existing stream definition are immediately addressed in the policy server through a new comprehensive policy that is then deployed to the switches. This tight integration not only allows the policy server to configure DAI, ACL and other mechanisms dynamically, but also to establish stream filtering and input gating based on 802.1Qci in order to achieve a robust and comprehensive security architecture.

The techniques disclosed herein enable the consumption of traffic intent from the well-known CNC, translation into new rulesets dynamically applied along the path, and the dynamic monitoring of the CNC rule sets for dynamic adaptation of rules as the traffic intent changes.

These techniques extend and expand concepts from one domain to another, and involve using the per-flow definitions that reside in the CNC and consolidating them with other security and QoS rules that are already owned by the policy server. As a result, the policy server not only includes the existing rules, but also creates new rules for the TSN flows dynamically generated by the CNC. To accomplish these functions, the policy server needs to understand the ranking and prioritization of the TSN flows and constructs a rule set that permits TSN flows and prevents other non-TSN traffic that attempts to communicate with the TSN endpoints. In effect, the dynamic 802.1Qci flows are used to create a QoS-aware TSN firewall.

The proposed techniques employ a centralized policy server that consolidates the per-flow security policy via 802.1Qci with the network-wide security policy and then deploys and manages the policies on the switches. To provision these policy rules, the policy server can communicate back the consolidated rule set to a provisioning tool, such as the CNC. According to the proposed techniques, the policy server, as an authorization tool, is to consume TSN-specific data and returns rules that become significant for the security of the TSN flows.

In summary, techniques are described herein provide a centralized policy management. A policy server is configured to combine and correlate the intent and context information retrieved from 802.1Qci traffic intent files, and to create a consistent and comprehensive policy along the intended path. The policy server is further configured to monitor the files present in these CNC components to dynamically absorb and parse new file versions and to adapt the corresponding policy to any change in the configuration of the automation system.