TVR (Time-Variant Routing) Applicability

draft-zdm-tvr-applicability-05

Authors

L. Zhang
J. Dong
Huawei

M. Boucadair
Orange

Contributors

D. King
C. Rotsos
Lancaster University

P. Liu
China Mobile

T. Li
Juniper Networks

Overview

Definition for TVR Applicability statement

This document should provide an applicability statement on **how** the information and **data models may be used**, along with required **ancillary IETF technology**, to solve the **use cases and requirements**.

Content of This Draft

- Applicability of the TVR Yang YANG Model: when and how to use the TVR data models;
- Time synchronization: necessary ancillary technologies
- Operational considerations: Considerations on the schedule dissemination, execution, recovery, and error handling
- Security considerations: Considerations on how to mitigated the potential attacks)
- Appendix A: Code examples for each examples introduced in RFC9567.

What’s new in version 05?

- Editorial/Readability Improvements.
- Explaining TVR definitions (managing device, network controller, managed devices)
- Clarification for protocol usage and data stores related to TVR usage
- Practical Operator Considerations
- More details on handling schedule conflicts
- Recommendations for Security Considerations
- Cleaning up code and NITS related to JSON

Detailed Updates 1/5

- Explaining TVR definitions.
  - ➢Managing Device
  - ➢Network Controller
  - ➢Managed Device

2. Conventions and Definitions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

This document uses the following terms:

Managing Device: A centralized entity responsible for generating and maintaining TVR schedules. The managing device distributes schedules to network controllers and/or managed devices using the TVR YANG modules. In some deployments, the managing device may also serve as the network controller.

Network Controller: An entity that receives topology schedules from the managing device and performs route computation based on time-variant network conditions. The controller then distributes routing results to managed devices.

* Managed Device: A network device (e.g., router, switch) that receives schedules and/or routing instructions, and executes them according to the specified time windows. Managed devices may receive schedules directly from the managing device or routing results from the network controller.

Detailed Updates 2/5

- Clarification for protocol usage and data stores related to TVR usage
  - Intended configuration datastore: used for schedule provision
  - Operational datastore: used to store execution status and applied schedules

3.3. Encoding of the YANG Model

The TVR data model [I-D.ietf-tvr-schedule-yang] can manage network resources and topologies with scheduled attributes. There are modules defined in the TVR data model, these are:

* The ~~"ietf-tvr-schedule"~~ "ietf-tvr-schedule" module contains the schedule YANG definitions. This module uses groupings from [I-D.ietf-netmod-schedule-yang] data model;

* The ~~"ietf-tvr-topology"~~ "ietf-tvr-topology" module defines a network topology with a time-variant availability schedule;

* The ~~"ietf-tvr-node"~~ "ietf-tvr-node" module is to be used to manage the scheduled attributes of a single node.

To create a schedule, the following TVR data model objects and subsequent branches are used:

* ~~'node-schedule'~~ 'node-schedule'

* ~~'interface-schedule'~~ 'interface-schedule'

* ~~'attribute-schedule'~~ 'attribute-schedule'

When using these YANG modules with NETCONF or RESTCONF, implementations SHOULD target the intended configuration datastore for schedule provisioning and MAY read from the operational datastore to retrieve execution status and applied schedules. Clients can use NMDA (Network Management Datastore Architecture, [RFC8342]) operations to distinguish between intended configuration and actual operational state. For example, the managing device writes schedules to the intended or running datastore, and network devices report execution status via the operational datastore.

Detailed Updates 3/5

- Practical Operator Considerations for Time Synchronization
  - Maximum acceptable time-error bound;

Different time-variant scenarios may require different granularities of time synchronization. For example, the period of traffic and topology changes in tidal networks is usually a day or week. Therefore, a second-level time synchronization is enough. However, for the dynamic reachability scenarios, a fine-granularity time synchronization may be necessary, as the nodes may moving very fast in some cases (the moving speed of a low earth orbit satellite is more than 7900 m/s)

Operators SHOULD derive a maximum acceptable time-error bound based on the schedule granularity, execution jitter tolerance, and activation window requirements. For instance, if a schedule has a 1-second activation window and the system can tolerate up to 100ms of execution jitter, the time synchronization error MUST be kept well below 900ms. The chosen time synchronization protocol and configuration MUST be capable of meeting this derived bound under all expected network conditions.

Detailed Updates 4/5

- More details on handling schedule conflicts
  - Schedule conflicts: retaining the last-known-good schedule.
  - Schedule updates: completely replacement, partial updates are not permitted.

6.4.1. Consistency Error

Consistency error means that some time parameters conflict with other time parameters in the same schedule or in other schedules.

* If the time parameters of a schedule conflict with each other, for example, the period-start ~~bigger~~ **later** than period-end, the duration is longer than the product of frequency and interval, or the duration is longer than utc-until, then the schedule should be discarded and an error should be returned to the ~~managed device.~~ **schedule originator (e.g., the managing device or management client).**

* If there is a conflict between schedules with different schedule IDs, for example, schedule1 indicates that interface B is closed at time A, but schedule2 indicates that interface B is open at time A, then ~~all~~ **only the** conflicting ~~schedules~~ **update** should be ~~discarded and an~~ **rejected (retaining the last-known-good schedule). An** error should be returned to the ~~managed device.~~ **schedule originator, and the conflict MUST be logged for audit purposes.** If two schedules have the same schedule ID, then it is considered as ~~a~~ **an** update of the former schedule.

~~Editor's Note: multi-manager scenarios need~~ **Updates with the same schedule ID SHALL completely replace the previous schedule (full replacement, not merge). Implementations SHOULD support versioning or etag-based mechanisms** to ~~be considered.~~ **detect concurrent updates. Partial updates to a schedule are NOT permitted; clients MUST send the complete schedule definition.**

Detailed Updates 5/5

- Recommendations for Security Considerations
  - Schedule tampering and malicious schedule injection
  - Approaches: authentication and authorization, integrity protection, audit logging , and rate limiting and anomaly detection.

7.7. Schedule Tampering and Malicious Schedule Injection

Unauthorized modification or injection of malicious TVR schedules poses significant operational and security risks. An attacker who successfully tampers with schedules could cause traffic blackholing (by scheduling link or node shutdowns at critical times), trigger costly network-wide reroutes, degrade service-level agreement (SLA) performance, or enable targeted interception of sensitive traffic flows. Such attacks undermine the predictability and reliability that TVR aims to provide.

Mitigating schedule tampering requires a defense-in-depth approach:

* Authentication and Authorization: All schedule updates MUST be authenticated to verify the identity of the originator. Role based access control (RBAC) or attribute-based access control (ABAC) SHOULD be enforced to ensure that only authorized entities can modify schedules.

* Integrity Protection: Schedules MUST be protected against tampering in transit and at rest using cryptographic integrity mechanisms (e.g., digital signatures, HMAC). NETCONF over TLS [RFC7589], RESTCONF over TLS, and similar secure transport protocols provide such protection.

* Audit Logging: All schedule creation, modification, and deletion operations SHOULD be logged with timestamps, originator identity, and a description of the change. These logs are essential for forensic analysis and detecting anomalous behavior.

* Rate Limiting and Anomaly Detection: Implementations SHOULD enforce rate limits on schedule update operations and deploy anomaly detection mechanisms to identify suspicious patterns (e.g., rapid schedule churn, schedules from unexpected sources).

Next Steps

- Repository: https://github.com/zhangli-abcd/TVR-Applicability
  - No open issues currently
- What's next?
  - WG adoption?
  - Continue to improve the document
    - But!
    - Should the **Operational** and **Security** content (update 3, 4 & 5) be moved to the Operational Considerations Charter item?