Stream: Internet Engineering Task Force (IETF)
RFC: 9922
Category: Standards Track
Published: February 2026
ISSN: 2070-1721
Authors: Q. Ma, Ed. *Huawei*; Q. Wu *Huawei*; M. Boucadair, Ed. *Orange*; D. King *Lancaster University*

RFC 9922
A Common YANG Data Model for Scheduling

Abstract

This document defines common types and groupings that are meant to be used for scheduling purposes, such as events, policies, services, or resources based on date and time. For the sake of better modularity, the YANG module includes a set of recurrence-related groupings with varying levels of representation (i.e., from basic to advanced) to accommodate a variety of requirements. It also defines groupings for validating requested schedules and reporting scheduling statuses.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9922.

Copyright Notice

Table of Contents

1. Introduction 3
2. Conventions and Definitions 4
3. Module Overview 5
   3.1. Features 5
   3.2. Types and Identities 5
   3.3. Scheduling Groupings 5
      3.3.1. The "generic-schedule-params" Grouping 6
      3.3.2. The "period-of-time" Grouping 7
      3.3.3. The "recurrence-basic" Grouping 8
      3.3.4. The "recurrence-utc" Grouping 8
      3.3.5. The "recurrence-with-time-zone" Grouping 9
      3.3.6. The "recurrence-utc-with-periods" Grouping 10
      3.3.7. The "recurrence-time-zone-with-periods" Grouping 11
      3.3.8. The "icalendar-recurrence" Grouping 11
      3.3.9. The "schedule-status", "schedule-status-with-time-zone", and "schedule-status-with-name" Groupings 13
   3.4. Features Use and Augmentations 15
4. Some Usage Restrictions 15
5. Relationship to the DISMAN-SCHEDULE-MIB 16
6. The "ietf-schedule" YANG Module 17
7. Security Considerations 32
8. IANA Considerations 33
   8.1. The IETF XML Registry 33
   8.2. The YANG Module Names Registry 33
9. References 33
   9.1. Normative References 33
   9.2. Informative References 34

Appendix A. Examples of Scheduling Format Representation 36
A.1. The "generic-schedule-params" Grouping 38
A.2. The "period-of-time" Grouping 39
A.3. The "recurrence-basic" Grouping 40
A.4. The "recurrence-utc" Grouping 40
A.5. The "recurrence-with-time-zone" Grouping 40
A.6. The "recurrence-utc-with-periods" Grouping 41
A.7. The "recurrence-time-zone-with-periods" Grouping 41
A.8. The "icalendar-recurrence" Grouping 42
A.9. The "schedule-status" Grouping 44
Appendix B. Examples of Using/Extending the "ietf-schedule" Module 45
B.1. Example: Schedule Tasks to Execute Based on a Recurrence Rule 45
B.2. Example: Schedule Network Properties to Change Based on Date and Time 47
Appendix C. Examples of Using the "ietf-schedule" Module for Scheduled Use of Resources Framework 50
Acknowledgments 51
Authors' Addresses 52

1. Introduction

This document defines a common schedule YANG module ("ietf-schedule") that can be used in several scheduling contexts, e.g., (but not limited to) [YANG-NAC], [YANG-OAM], and [YANG-SCHEDULE]. The module includes a set of reusable groupings that are designed to be applicable for scheduling purposes, such as events, policies, services, or resources based on date and time. It also defines groupings for validating requested schedules and reporting scheduling statuses.

This document does not make any assumption about the nature of actions that are triggered by the schedules. Detection and resolution of any schedule conflicts are beyond the scope of this document.

Section 5 discusses the relationship with the Management Information Base (MIB) managed objects for scheduling management operations defined in [RFC3231].

Appendix A describes a set of examples to illustrate the use of the common schedule groupings (Section 3.3). Appendix B provides sample modules to exemplify how future modules can use the extensibility provisions in the "ietf-schedule" module (Section 6). Also, Appendix C provides an example of using the "ietf-schedule" module for scheduled use of a resources framework (e.g., [RFC8413]).

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.

The meanings of the symbols in tree diagrams are defined in [RFC8340].

This document uses the YANG terminology defined in Section 3 of [RFC7950].

This document makes use of the following terms:

Recurrence rule: Refers to a rule or repeating pattern for recurring events. See also Section 3.8.5.3 of [RFC5545] for a comprehensive iCalendar recurrence rule specification.

Recurrence instance (or Recurrence, for short): Refers to an instance that matches a recurrence rule.

Recurrence set: Refers to a set of recurrence instances.

Frequency: Characterizes the type of a recurrence rule. Values are taken from the "FREQ" rule in Section 3.3.10 of [RFC5545].

For example, repeating events based on an interval of a second or more are classified as recurrence with a frequency value of "SECONDLY". Frequency values defined as identities in the YANG module are used in lowercase.

iCalendar: Refers to Internet Calendaring per [RFC5545].

Interval: Refers to an integer that specifies the interval at which a recurrence rule repeats. Values are taken from the "INTERVAL" rule in Section 3.3.10 of [RFC5545].

For example, "1" means every second for a secondly rule, every minute for a minutely rule, every hour for an hourly rule, etc.

System: Refers to an entity that hosts a schedule that is managed using the YANG module defined in this document.

3. Module Overview

3.1. Features

The "ietf-schedule" data model defines the recurrence-related groupings using a modular approach. To that aim, a variety of representations of recurrence groupings ranging from basic to advanced (iCalender-like) are defined. To allow for different options, two features are defined in the data model:

- "basic-recurrence"
- "icalendar-recurrence"

Refer to Section 3.4 and Appendix B.1 for the use of these features.

3.2. Types and Identities

The "ietf-schedule" module (Section 6) defines the following identities:

- "schedule-type": Indicates the type of schedule. The following types are defined so far:
  - one-shot: The schedule will trigger an action that has either the duration specified as 0 or the end time specified as the same as the start time, and then the schedule will disable itself (Section 3.3 of [RFC3231]).
  - period: This type is used for a period-based schedule consisting of either (1) a start and end or (2) a start and positive duration of time. If neither an end nor a duration is indicated, the period is considered to last forever.
  - recurrence: This type is used for a recurrence-based schedule. A recurrence may be periodic (i.e., repeat over the same period, e.g., every five minutes) or not (i.e., repeat in a non-regular manner, e.g., every day at 8 and 9 AM).
- "frequency-type": Characterizes the repeating interval rule of a recurrence schedule (secondly, minutely, etc.).
- "schedule-state": Indicates the status of a schedule (enabled, disabled, conflicted, finished, etc.). This identity can also be used to manage the state of individual instances of a recurrence-based schedule.
- "discard-action-type": Specifies the action for the responder to take (e.g., generate a warning or an error message) when a requested schedule cannot be accepted for any reason and is discarded.

3.3. Scheduling Groupings

The "ietf-schedule" module (Section 6) defines the following groupings:

- "generic-schedule-params" (Section 3.3.1)
- "period-of-time" (Section 3.3.2)

- "recurrence-basic" (Section 3.3.3)
- "recurrence-utc" (Section 3.3.4)
- "recurrence-with-time-zone" (Section 3.3.5)
- "recurrence-utc-with-periods" (Section 3.3.6)
- "recurrence-time-zone-with-periods" (Section 3.3.7)
- "icalendar-recurrence" (Section 3.3.8)
- "schedule-status", "schedule-status-with-time-zone", and "schedule-status-with-name" (Section 3.3.9)

Examples are provided in Appendix A.

3.3.1. The "generic-schedule-params" Grouping

A system accepts and handles schedule requests, which may help further automate the scheduling process of events, policies, services, or resources based on date and time. The "generic-schedule-params" grouping (Figure 1) specifies a set of configuration parameters that are used by a system for validating requested schedules.

```
grouping generic-schedule-params:
  +-- description?            string
  +-- time-zone-identifier?   sys:timezone-name
  +-- validity?               yang:date-and-time
  +-- max-allowed-start?      yang:date-and-time
  +-- min-allowed-start?      yang:date-and-time
  +-- max-allowed-end?        yang:date-and-time
  +-- discard-action?         identityref
```

Figure 1: 'generic-schedule-params' Grouping Tree Structure

The "description" parameter includes a description of the schedule. No constraint is imposed on the structure nor the use of this parameter.

The "time-zone-identifier" parameter, if provided, specifies the time zone reference [RFC7317] of the local date and time values. This parameter **MUST** be specified if any of the date and time values are in the format of local time. It **MUST NOT** be applied to date and time values that are specified in the format of UTC or time zone offset to UTC.

The "validity" parameter specifies the date and time after which a schedule will not be considered as valid. It determines the latest time that a schedule can be started to execute independent of when it ends, and it takes precedence over similar attributes that are provided at the schedule instance itself. A requested schedule may still be accepted, but any occurrences that start later than the configured value will not be executed.

The "max/min-allowed-start" parameters specify the maximum/minimum scheduled start date and time. A requested schedule will be rejected if the first occurrence of the schedule starts later/earlier than the configured values.

The "max-allowed-end" parameter specifies the maximum allowed end time of the last occurrence. A requested schedule will be rejected if the end time of the last occurrence is later than the configured "max-allowed-end" value.

The "discard-action" parameter specifies the action if a requested schedule cannot be accepted for any reason and is discarded. Possible reasons include, but are not limited to, the requested schedule failing to satisfy the guards in this grouping, conflicting with existing schedules, or being out-of-date (e.g., the expected start has already passed).

*

These parameters apply to all schedules on a system and are meant to provide guards against stale configuration, too short schedule requests that would prevent validation by admins of some critical systems, etc.

3.3.2. The "period-of-time" Grouping

The "period-of-time" grouping (Figure 2) represents a time period using either a start date and time ("period-start") and end date and time ("period-end") or a start date and time ("period-start") and a non-negative time duration ("duration"). For the first format, the start of the period **MUST** be no later than the end of the period. If neither an end date and time ("period-end") nor a duration ("duration") is indicated, the period is considered to last forever. If the duration ("duration") value is 0 or the end time ("period-end") is the same as the start time ("period-start"), the period is considered as a one-shot schedule. If no start date and time ("period-start") is specified, the period is considered to start immediately.

The "time-zone-identifier" parameter indicates the identifier for the time zone. This parameter **MUST** be specified if either the "period-start" or "period-end" value is reported in local time format. It **MUST NOT** be applied to date and time values that are specified in the format of UTC or time zone offset to UTC.

The "period-description" parameter includes a description of the period. No constraint is imposed on the structure nor the use of this parameter.

```
grouping period-of-time:
  +-- period-description?     string
  +-- period-start?           yang:date-and-time
  +-- time-zone-identifier?   sys:timezone-name
  +-- (period-type)?
     +--:(explicit)
     |  +-- period-end?       yang:date-and-time
     +--:(duration)
        +-- duration?         duration
```

Figure 2: 'period-of-time' Grouping Tree Structure

3.3.3. The "recurrence-basic" Grouping

The "recurrence-basic" grouping (Figure 3) specifies a simple recurrence rule that starts immediately and repeats forever.

```
grouping recurrence-basic:
  +-- recurrence-description?   string
  +-- frequency?                identityref
  +-- interval?                 uint32
```

Figure 3: 'recurrence-basic' Grouping Tree Structure

The frequency parameter ("frequency") identifies the type of recurrence rule. For example, a "daily" frequency value specifies repeating events based on an interval of a day or more.

Consistent with Section 3.3.10 of [RFC5545], the interval parameter ("interval") represents at which interval the recurrence rule repeats. For example, within a "daily" recurrence rule, an interval value of "8" means every eight days.

Note that, per Section 4.13 of [YANG-GUIDE], neither a "default" nor a "mandatory" substatement is defined here for both "frequency" and "interval" parameters because there are cases (e.g., profiling) where using these statements is problematic. YANG modules using this grouping **SHOULD** refine these two nodes with either a "mandatory" or a "default" statement if they always need to be configured or have default values. This recommendation **MAY** be ignored in cases such as when this grouping is used by another grouping.

The "recurrence-description" parameter includes a description of the period. No constraint is imposed on the structure nor the use of this parameter.

3.3.4. The "recurrence-utc" Grouping

The "recurrence-utc" grouping (Figure 4) uses the "recurrence-basic" grouping (Section 3.3.3) and specifies a simple recurrence rule in UTC format.

```
grouping recurrence-utc:
  +-- recurrence-first
  |  +-- start-time-utc?   yang:date-and-time
  |  +-- duration?         uint32
  +-- (recurrence-end)?
  |  +--:(until)
  |  |  +-- utc-until?          yang:date-and-time
  |  +--:(count)
  |     +-- count?              uint32
  +-- recurrence-description?   string
  +-- frequency?                identityref
  +-- interval?                 uint32
```

Figure 4: 'recurrence-utc' Grouping Tree Structure

The "start-time-utc" parameter indicates the start time in UTC format.

The "duration" parameter specifies, in units of seconds, the time period of the first occurrence. Unless specified otherwise (e.g., through additional augmented parameters), the "duration" also applies to subsequent recurrence instances. When unspecified, each occurrence is considered as immediate completion (e.g., execute an immediate command that is considered to complete quickly) or hard to compute an exact duration (e.g., run a data analysis script whose execution time may depend on the data volume and computation resource availability). The behavior to follow when a task takes more time than specified by the "duration" is out of scope. Such considerations belong to task management, not schedule management.

Note that the "interval" and "duration" cover two distinct properties of a schedule event. The interval specifies when a schedule will occur, combined with the frequency parameter, while the duration indicates how long an occurrence will last. This document allows the interval between occurrences to be shorter than the duration of each occurrence (e.g., a recurring event is scheduled to start every day for a duration of 2 days).

The repetition can be scoped by a specified end time or by a count of occurrences, indicated by the "recurrence-end" choice. The "count" value **MUST** be greater than 1, and the "start-time-utc" value always counts as the first occurrence.

The "recurrence-utc" grouping is designed to be reused in scheduling contexts where machine readability is more desirable.

3.3.5. The "recurrence-with-time-zone" Grouping

The "recurrence-with-time-zone" grouping (Figure 5) uses the "recurrence-basic" grouping (Section 3.3.3) and specifies a simple recurrence rule with a time zone.

```
grouping recurrence-with-time-zone:
  +-- recurrence-first
  |  +-- start-time?   yang:date-and-time
  |  +-- duration?     duration
  +-- time-zone-identifier?     sys:timezone-name
  +-- (recurrence-end)?
  |  +--:(until)
  |  |  +-- until?              yang:date-and-time
  |  +--:(count)
  |     +-- count?              uint32
  +-- recurrence-description?   string
  +-- frequency?                identityref
  +-- interval?                 uint32
```

Figure 5: 'recurrence-with-time-zone' Grouping Tree Structure

The "recurrence-first" container includes "start-time" and "duration" parameters to specify the start time and period of the first occurrence. Unless specified otherwise (e.g., through additional augmented parameters), the "duration" also applies to subsequent recurrence instances. When unspecified, each occurrence is considered as immediate completion (e.g., execute an immediate

command that is considered to complete quickly) or hard to compute an exact duration (e.g., run a data analysis script whose execution time may depend on the data volume and computation resource availability).

The grouping also includes a "time-zone-identifier" parameter, which **MUST** be specified if either the "start-time" or "until" value is reported in local time format. It **MUST NOT** be applied to date and time values that are specified in the format of UTC or time zone offset to UTC.

The repetition can be scoped by a specified end time or by a count of occurrences, indicated by the "recurrence-end" choice. The "count" value **MUST** be greater than 1, and the "start-time" value always counts as the first occurrence.

The considerations discussed in Section 3.3.4 for "interval" and "duration" are also applicable to "recurrence-with-time-zone".

Unlike the definition of the "recurrence-utc" grouping (Section 3.3.4), "recurrence-with-time-zone" is intended to promote human readability over machine readability.

3.3.6. The "recurrence-utc-with-periods" Grouping

The "recurrence-utc-with-periods" grouping (Figure 6) uses the "recurrence-utc" grouping (Section 3.3.4) and adds a "period-timeticks" list to define an aggregate set of repeating occurrences.

```
grouping recurrence-utc-with-periods:
  +-- recurrence-first
  |  +-- start-time-utc?   yang:date-and-time
  |  +-- duration?         uint32
  +-- (recurrence-end)?
  |  +--:(until)
  |  |  +-- utc-until?          yang:date-and-time
  |  +--:(count)
  |     +-- count?              uint32
  +-- recurrence-description?   string
  +-- frequency?                identityref
  +-- interval?                 uint32
  +-- period-timeticks* [period-start]
     +-- period-start    yang:timeticks
     +-- period-end?     yang:timeticks
```

Figure 6: 'recurrence-utc-with-periods' Grouping Tree Structure

The recurrence instances are specified by the union of occurrences defined by both the recurrence rule and "period-timeticks" list. This list uses the "yang:timeticks" type defined in [RFC6991]. Duplicate instances are ignored. The value of the "period-start" instance **MUST NOT** exceed the value indicated by the value of the "frequency" instance, i.e., the "timeticks" value must not exceed 100 in a secondly recurrence rule, and it must not exceed 6000 in a minutely recurrence rule, and so on.

3.3.7. The "recurrence-time-zone-with-periods" Grouping

The "recurrence-time-zone-with-periods" grouping (Figure 7) uses the "recurrence-with-time-zone" grouping (Section 3.3.5) and adds a "period" list to define an aggregate set of repeating occurrences.

```
grouping recurrence-time-zone-with-periods:
  +-- recurrence-first
  |  +-- start-time?   yang:date-and-time
  |  +-- duration?     duration
  +-- time-zone-identifier?     sys:timezone-name
  +-- (recurrence-end)?
  |  +--:(until)
  |  |  +-- until?              yang:date-and-time
  |  +--:(count)
  |     +-- count?              uint32
  +-- recurrence-description?   string
  +-- frequency?                identityref
  +-- interval?                 uint32
  +-- period* [period-start]
     +-- period-description?     string
     +-- period-start            yang:date-and-time
     +-- time-zone-identifier?   sys:timezone-name
     +-- (period-type)?
        +--:(explicit)
        |  +-- period-end?       yang:date-and-time
        +--:(duration)
           +-- duration?         duration
```

Figure 7: 'recurrence-time-zone-with-periods' Grouping Tree Structure

The recurrence instances are specified by the union of occurrences defined by both the recurrence rule and "period" list. Duplicate instances are ignored.

3.3.8. The "icalendar-recurrence" Grouping

The "icalendar-recurrence" grouping (Figure 8) uses the "recurrence-time-zone-with-periods" grouping (Section 3.3.7) and defines more data nodes to enrich the definition of recurrence. The structure of the "icalendar-recurrence" grouping refers to the definition of the recurrence component defined in Sections 3.3.10 and 3.8.5 of [RFC5545].

```
grouping icalendar-recurrence:
  +-- recurrence-first
  |  +-- start-time?   yang:date-and-time
  |  +-- duration?     duration
  +-- time-zone-identifier?     sys:timezone-name
  +-- (recurrence-end)?
  |  +--:(until)
  |  |  +-- until?              yang:date-and-time
  |  +--:(count)
  |     +-- count?              uint32
  +-- recurrence-description?   string
  +-- frequency?                identityref
  +-- interval?                 uint32
  +-- period* [period-start]
  |  +-- period-description?     string
  |  +-- period-start            yang:date-and-time
  |  +-- time-zone-identifier?   sys:timezone-name
  |  +-- (period-type)?
  |     +--:(explicit)
  |     |  +-- period-end?       yang:date-and-time
  |     +--:(duration)
  |        +-- duration?         duration
  +-- bysecond*                 uint32
  +-- byminute*                 uint32
  +-- byhour*                   uint32
  +-- byday* [weekday]
  |  +-- direction*   int32
  |  +-- weekday      schedule:weekday
  +-- bymonthday*               int32
  +-- byyearday*                int32
  +-- byyearweek*               int32
  +-- byyearmonth*              uint32
  +-- bysetpos*                 int32
  +-- workweek-start?           schedule:weekday
  +-- exception-dates*          yang:date-and-time
```

Figure 8: 'icalendar-recurrence' Grouping Tree Structure

An array of the "bysecond" (or "byminute" or "byhour") specifies a list of seconds within a minute (or minutes within an hour or hours of the day). For example, within a "minutely" recurrence rule, the values of "byminute" node "10" and "20" mean the occurrences are generated at the 10th and 20th minute within an hour, reducing the number of recurrence instances from all minutes.

The parameter "byday" specifies a list of days of the week, with an optional direction that indicates the nth occurrence of a specific day within the "monthly" or "yearly" frequency instance. Valid values of "direction" are 1 to 5 or -5 to -1 within a "monthly" recurrence rule and 1 to 53 or -53 to -1 within a "yearly" recurrence rule. For example, within a "monthly" rule, the "weekday" with a value of "monday" and the "direction" with a value of "-1" represents the last Monday of the month.

An array of the "bymonthday" (or byyearday", "byyearweek", or "byyearmonth") specifies a list of days of the month (or days of the year, weeks of the year, or months of the year). For example, within a "yearly" recurrence rule, the values of "byyearmonth" instances "1" and "2" mean the occurrences are generated in January and February, increasing the "yearly" recurrence from every year to every January and February of the year.

The "bysetpos" conveys a list of values that corresponds to the nth occurrence within the set of recurrence instances to be specified. For example, in a "monthly" recurrence rule, the "byday" data node specifies every Monday of the week, and the "bysetpos" with a value of "-1" represents the last Monday of the month. Not setting the "bysetpos" data node represents every Monday of the month.

The "workweek-start" data node specifies the day on which the week starts. This is significant when a "weekly" recurrence rule has an interval greater than 1, and a "byday" data node is specified. This is also significant when in a "yearly" rule and a "byyearweek" is specified. Note that, per Section 4.13 of [YANG-GUIDE], neither a "default" nor a "mandatory" substatement is defined here because there are cases (e.g., profiling) where using these statements is problematic. YANG modules using this grouping **SHOULD** refine the "workweek-start" node with either a "mandatory" or a "default" statement if it always needs to be configured or has a default value. This **MAY** be ignored in cases such as when this grouping is used by another grouping.

The "exception-dates" data node specifies a list of exceptions for recurrence. The final recurrence set is generated by gathering all of the date and time values created by any of the specified recurrence rules and date-times and then excluding any start date and time values specified by "exception-dates" parameter.

3.3.9. The "schedule-status", "schedule-status-with-time-zone", and "schedule-status-with-name" Groupings

The "schedule-status", "schedule-status-with-time-zone", and "schedule-status-with-name" groupings (Figure 9) define common parameters for scheduling management/status exposure. The "schedule-status-with-time-zone" grouping has the same structure as "schedule-status" but with an additional parameter to identify a time zone. Similarly, the "schedule-status-with-name" grouping has the same structure as "schedule-status" but with an additional parameter to identify a schedule "schedule-name". These structures are defined in the module to allow for better modularity and flexibility.

```
grouping schedule-status:
  +-- state?                     identityref
  +-- version?                   uint16
  +-- schedule-type?             identityref
  +--ro local-time?                yang:date-and-time
  +--ro last-update?               yang:date-and-time
  +--ro counter?                   yang:counter32
  +--ro last-occurrence?           yang:date-and-time
  +--ro upcoming-occurrence?       yang:date-and-time
  +--ro last-failed-occurrence?    yang:date-and-time
  +--ro failure-counter?           yang:counter32
grouping schedule-status-with-time-zone:
  +--ro time-zone-identifier?  sys:timezone-name
  +-- schedule-name?             string
  +-- state?                     identityref
  +-- version?                   uint16
  +-- schedule-type?             identityref
  +--ro local-time?                yang:date-and-time
  +--ro last-update?               yang:date-and-time
  +--ro counter?                   yang:counter32
  +--ro last-occurrence?           yang:date-and-time
  +--ro upcoming-occurrence?       yang:date-and-time
  +--ro last-failed-occurrence?    yang:date-and-time
  +--ro failure-counter?           yang:counter32
grouping schedule-status-with-name:
  +-- schedule-name?             string
  +-- state?                     identityref
  +-- version?                   uint16
  +-- schedule-type?             identityref
  +--ro local-time?                yang:date-and-time
  +--ro last-update?               yang:date-and-time
  +--ro counter?                   yang:counter32
  +--ro last-occurrence?           yang:date-and-time
  +--ro upcoming-occurrence?       yang:date-and-time
  +--ro last-failed-occurrence?    yang:date-and-time
  +--ro failure-counter?           yang:counter32
```

Figure 9: 'schedule-status-' Groupings Tree Structure*

The "schedule-name" parameter is useful to uniquely identify a schedule in a network device or controller if multiple scheduling contexts exist.

The "state" parameter is defined to configure/expose the scheduling state, depending on the use of the grouping. For a recurrence-based schedule, it represents the state of the overall recurrence. The "identityref" type is used for this parameter to allow extensibility in future modules.

The "version" parameter is used to track the current schedule version information. The version can be incremented by the entity that created the schedule. The "last-update" parameter identifies when the schedule was last modified. In some contexts, this parameter can be used to track the configuration of a given schedule. In such cases, the "version" may not be used.

The "schedule-type" parameter identifies the type of the current schedule. The "counter", "last-occurrence", and "upcoming-occurrence" data nodes are only available when the "schedule-type" is "recurrence".

When no time zone is included, "local-time" reports the actual local time as seen by the entity that hosts a schedule. This parameter can be used by a controller to infer the offset to UTC. This use is similar to the use of "schedLocalTime" in [RFC3231].

"last-failed-occurrence" and "failure-counter" report the last failure that occurred and the count of failures for this schedule. Unless new parameters/operations are defined to allow the count of failures to be reset, "failure-counter" is reset by default only when the schedule starts.

The current groupings capture common parameters that are applicable to typical scheduling contexts known so far. Future modules can define other useful parameters as needed. For example, in a scheduling context with multiple system sources to feed the schedules, the "source" and "precedence" parameters may be needed to reflect how schedules from different sources should be prioritized.

3.4. Features Use and Augmentations

Appendix B.1 provides an example about how the features defined in Section 3.1 can be used. Implementations may support a basic recurrence rule or an advanced one, as needed, by declaring different features. Whether only one or both features are supported is implementation specific and depends on the specific scheduling context.

The common schedule groupings (Section 3.3) can also be augmented to support specific needs. As an example, Appendix B.2 demonstrates how additional parameters can be added to comply with specific schedule needs.

4. Some Usage Restrictions

There are some restrictions that need to be followed when using groupings defined in the "ietf-schedule" YANG module (Section 3.3):

- The instant in time represented by "period-start" **MUST** be before the "period-end" for the "period-of-time" grouping (Section 3.3.2).
- The combination of the day, month, and year represented for date and time values **MUST** be valid. See Section 5.7 of [RFC3339] for the maximum day number based on the month and year.
- Unless deployed in contexts where time synchronization is not subject to leap second adjustments (e.g., Section 4.3 of [NTPv5]), the second for date and time values **SHOULD** have the value "60" at the end of months in which a leap second occurs.

- Schedules received with a starting time in the past with respect to current time **SHOULD** be ignored. When a local policy is provided, an implementation **MAY** omit the past occurrences and start immediately (e.g., for a period-based schedule) or start from the date and time when the recurrence pattern is first satisfied from the current time (e.g., for a recurrence-based schedule).

5. Relationship to the DISMAN-SCHEDULE-MIB

[RFC3231] specifies a Management Information Base (MIB) used to schedule management operations periodically or at specified dates and times.

Although no data nodes are defined in this document, Table 1 lists how the main objects in the DISMAN-SCHEDULE-MIB can be mapped to YANG parameters.

MIB Object	YANG
schedLocalTime	local-time
schedType	schedule-type
schedName	schedule-name
schedOwner	Not Supported
schedDescr	description
schedInterval	interval
schedWeekDay	weekday
schedMonth	byyearmonth
schedDay	bymonthday
schedHour	byhour
schedMinute	byminute
schedContextName	Not Supported
schedAdminStatus	state
schedOperStatus	state
schedFailures	failure-counter
schedLastFailure	Not Supported

MIB Object	YANG
schedLastFailed	last-failed-occurrence
schedStorageType	Not Supported
schedVariable	Not applicable
schedValue	Not applicable
schedTriggers	counter/failure-counter

Table 1: YANG/MIB Mapping

6. The "ietf-schedule" YANG Module

This module imports types defined in [RFC5545], [RFC6991], and [RFC7317].

```
<CODE BEGINS>
file "ietf-schedule@2026-02-18.yang"
module ietf-schedule {
  yang-version 1.1;
  namespace "urn:ietf:params:xml:ns:yang:ietf-schedule";
  prefix schedule;

  import ietf-yang-types {
    prefix yang;
    reference
      "RFC 6991: Common YANG Data Types";
  }

  import ietf-system {
    prefix sys;
    reference
      "RFC 7317: A YANG Data Model for System Management";
  }

  organization
    "IETF NETMOD Working Group";
  contact
    "WG Web:  <https://datatracker.ietf.org/wg/netmod/>
     WG List: <mailto:netmod@ietf.org>

     Editor:   Qiufang Ma
               <mailto:maqiufang1@huawei.com>
     Author:   Qin Wu
               <mailto:bill.wu@huawei.com>
     Editor:   Mohamed Boucadair
               <mailto:mohamed.boucadair@orange.com>
     Author:   Daniel King
               <mailto:d.king@lancaster.ac.uk>";
  description
    "This YANG module defines a set of common types and groupings
     that are applicable for scheduling purposes, such as events,
```

```
     policies, services, or resources based on date and time.

     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 (RFC 2119) (RFC 8174) when, and only when,
     they appear in all capitals, as shown here.

     Copyright (c) 2026 IETF Trust and the persons identified
     as authors of the code.  All rights reserved.

     Redistribution and use in source and binary forms, with
     or without modification, is permitted pursuant to, and
     subject to the license terms contained in, the Revised
     BSD License set forth in Section 4.c of the IETF Trust's
     Legal Provisions Relating to IETF Documents
     (https://trustee.ietf.org/license-info).

     This version of this YANG module is part of RFC 9922; see
     the RFC itself for full legal notices.

     All revisions of IETF and IANA-maintained modules can be found
     in the 'YANG Parameters' registry group
     (https://www.iana.org/assignments/yang-parameters).";

   revision 2026-02-18 {
     description
       "Initial revision.";
     reference
       "RFC 9922: A Common YANG Data Model for Scheduling";
   }

   feature basic-recurrence {
     description
       "Indicates that the server supports configuring a basic
        scheduled recurrence.";
   }

   feature icalendar-recurrence {
     description
       "Indicates that the server supports configuring a comprehensive
        scheduled iCalendar recurrence.";
     reference
       "RFC 5545: Internet Calendaring and Scheduling Core Object
                  Specification (iCalendar),
                  Sections 3.3.10 and 3.8.5";
   }

   typedef weekday {
     type enumeration {
       enum sunday {
         value 0;
         description
           "Sunday of the week.";
       }
       enum monday {
         value 1;
         description
```

```
          "Monday of the week.";
      }
      enum tuesday {
        value 2;
        description
          "Tuesday of the week.";
      }
      enum wednesday {
        value 3;
        description
          "Wednesday of the week.";
      }
      enum thursday {
        value 4;
        description
          "Thursday of the week.";
      }
      enum friday {
        value 5;
        description
          "Friday of the week.";
      }
      enum saturday {
        value 6;
        description
          "Saturday of the week.";
      }
    }
    description
      "Seven days of the week.";
  }

  typedef duration {
    type string {
      pattern '((\+)?|\-)P((([0-9]+)D)?(T(0[0-9]|1[0-9]|2[0-3])'
            + ':[0-5][0-9]:[0-5][0-9]))|P([0-9]+)W';
    }
    description
      "Duration of the time.  The format can represent nominal
       durations (weeks designated by 'W' and days designated by 'D')
       and accurate durations (hours:minutes:seconds follows the
       designator 'T').

       Note that this value type doesn't support the 'Y' and 'M'
       designators to specify durations in terms of years and months.

       Negative durations are typically used to schedule an alarm to
       trigger before an associated time.";
    reference
      "RFC 5545: Internet Calendaring and Scheduling Core Object
                 Specification (iCalendar), Sections 3.3.6 and
                 3.8.6.3";
  }

  identity schedule-type {
    description
      "Base identity for schedule type.";
  }
```

```
identity one-shot {
  base schedule-type;
  description
    "Indicates a one-shot schedule.  That is a schedule that
     will trigger an action with the duration being specified as
     0 or end time being specified as the same as the start time,
     and then the schedule will disable itself.";
}

identity period {
  base schedule-type;
  description
    "Indicates a period-based schedule consisting of either a
     start and end or a start and positive duration of time.  If
     neither an end nor a duration is indicated, the period is
     considered to last forever.";
}

identity recurrence {
  base schedule-type;
  description
    "Indicates a recurrence-based schedule.";
}

identity frequency-type {
  description
    "Base identity for frequency type.";
}

identity secondly {
  base frequency-type;
  description
    "Indicates a repeating rule based on an interval of
     a second or more.";
}

identity minutely {
  base frequency-type;
  description
    "Indicates a repeating rule based on an interval of
     a minute or more.";
}

identity hourly {
  base frequency-type;
  description
    "Indicates a repeating rule based on an interval of
     an hour or more.";
}

identity daily {
  base frequency-type;
  description
    "Indicates a repeating rule based on an interval of
     a day or more.";
}
```

```
  identity weekly {
    base frequency-type;
    description
      "Indicates a repeating rule based on an interval of
       a week or more.";
  }

  identity monthly {
    base frequency-type;
    description
      "Indicates a repeating rule based on an interval of
       a month or more.";
  }

  identity yearly {
    base frequency-type;
    description
      "Indicates a repeating rule based on an interval of
       a year or more.";
  }

  identity schedule-state {
    description
      "Base identity for schedule state.";
  }

  identity enabled {
    base schedule-state;
    description
      "Indicates a schedule with an enabled state.";
  }

  identity finished {
    base schedule-state;
    description
      "Indicates a schedule with a finished state.
       The finished state indicates that the schedule has ended.";
  }

  identity disabled {
    base schedule-state;
    description
      "Indicates a schedule with a disabled state.";
  }

  identity out-of-date {
    base schedule-state;
    description
      "Indicates a schedule that is received out-of-date.";
  }

  identity conflicted {
    base schedule-state;
    description
      "Indicates a schedule with a conflicted state with other
       schedules.";
  }
```

```
  identity discard-action-type {
    description
      "Base identity for the action for the responder to take
       when a requested schedule cannot be accepted for any
       reason and is discarded.";
  }

  identity warning {
    base discard-action-type;
    description
      "Indicates that a warning message is generated
       when a schedule is discarded.";
  }

  identity error {
    base discard-action-type;
    description
      "Indicates that an error message is generated
       when a schedule is discarded.";
  }

  identity silently-discard {
    base discard-action-type;
    description
      "Indicates that a schedule that is not valid is silently
       discarded.";
  }

  grouping generic-schedule-params {
    description
      "Includes a set of generic parameters that are followed by
       the entity that supports schedules.

       Such parameters are used as guards to prevent, e.g., stale
       configuration.";
    leaf description {
      type string;
      description
        "Provides a description of the schedule.";
    }
    leaf time-zone-identifier {
      type sys:timezone-name;
      description
        "Indicates the identifier for the time zone.  This parameter
         MUST be specified if any of the date and time values are
         in the format of local time.  It MUST NOT be applied to
         date and time values that are specified in the format of
         UTC or time zone offset to UTC.";
    }
    leaf validity {
      type yang:date-and-time;
      description
        "Specifies the date and time after which a schedule will not
         be considered as valid.  This parameter takes precedence
         over similar attributes that are provided at the schedule
         instance itself.";
    }
    leaf max-allowed-start {
```

```
      type yang:date-and-time;
      description
        "Specifies the maximum scheduled start date and time.
         A requested schedule whose first instance occurs after
         this value cannot be accepted by the entity.  Specifically,
         a requested schedule will be rejected if the first
         occurrence of that schedule exceeds 'max-allowed-start'.";
    }
    leaf min-allowed-start {
      type yang:date-and-time;
      description
        "Specifies the minimum scheduled start date and time.
         A requested schedule whose first instance occurs before
         this value cannot be accepted by the entity.  Specifically,
         a requested schedule will be rejected if the first
         occurrence of that schedule is scheduled before
         'min-allowed-start'.";
    }
    leaf max-allowed-end {
      type yang:date-and-time;
      description
        "A requested schedule will be rejected if the end time of
         the last occurrence exceeds 'max-allowed-end'.";
    }
    leaf discard-action {
      type identityref {
        base discard-action-type;
      }
      description
        "Specifies the behavior when a schedule is discarded for
         any reason, e.g., failing to satisfy the guards in this
         grouping or being received out-of-date.";
    }
  }

  grouping period-of-time {
    description
      "This grouping is defined for the period of time property.";
    reference
      "RFC 5545: Internet Calendaring and Scheduling Core Object
                 Specification (iCalendar), Section 3.3.9";
    leaf period-description {
      type string;
      description
        "Provides a description of the period.";
    }
    leaf period-start {
      type yang:date-and-time;
      description
        "Period start time.";
    }
    leaf time-zone-identifier {
      type sys:timezone-name;
      description
        "Indicates the identifier for the time zone.  This parameter
         MUST be specified if either the 'period-start' or
         'period-end' value is reported in local time format.
         It MUST NOT be applied to date and time values that are
```

```
          specified in the format of UTC or time zone offset
          to UTC.";
      }
      choice period-type {
        description
          "Indicates the type of the time period.  Two types are
           supported.  If no choice is indicated, the period is
           considered to last forever.";
        case explicit {
          description
            "A period of time is identified by its start and its end.
             'period-start' indicates the period start.";
          leaf period-end {
            type yang:date-and-time;
            description
              "A period of time is defined by a start and end time.
               The start MUST be no later than the end.  The period
               is considered as a one-shot schedule if the end time
               is the same as the start time.";
          }
        }
        case duration {
          description
            "A period of time is defined by a start and a non-negative
             duration of time.";
          leaf duration {
            type duration {
              pattern 'P((([0-9]+)D)?(T(0[0-9]|1[0-9]|2[0-3])'
                    + ':[0-5][0-9]:[0-5][0-9]))|P([0-9]+)W';
            }
            description
              "A non-negative duration of time.  This value is
               equivalent to the format of 'duration' type except that
               the value cannot be negative.  The period is considered
               to be a one-shot schedule if the value is 0.";
          }
        }
      }
    }

    grouping recurrence-basic {
      description
        "A simple definition of recurrence.";
      leaf recurrence-description {
        type string;
        description
          "Provides a description of the recurrence.";
      }
      leaf frequency {
        type identityref {
          base frequency-type;
        }
        description
          "Specifies the frequency type of the recurrence rule.";
      }
      leaf interval {
        type uint32 {
          range "1..max";
```

```
      }
      must '../frequency' {
        error-message "Frequency must be provided.";
      }
      description
        "A positive integer representing the interval at which the
         recurrence rule repeats.  For example, within a 'daily'
         recurrence rule, a value of '8' means every eight days.";
    }
  }

  grouping recurrence-utc {
    description
      "A simple definition of recurrence with time specified in
       UTC format.";
    container recurrence-first {
      description
        "Specifies the first instance of the recurrence.  If
         unspecified, the recurrence is considered to start from
         the date and time when the recurrence pattern is first
         satisfied.";
      leaf start-time-utc {
        type yang:date-and-time;
        description
          "Defines the date and time of the first instance
           in the recurrence set.  A UTC format MUST be used.";
      }
      leaf duration {
        type uint32;
        units "seconds";
        description
          "When specified, it indicates how long the first occurrence
           lasts.  Unless specified otherwise, it also applies to all
           the other instances in the recurrence set.";
      }
    }
    choice recurrence-end {
      description
        "Modes to control the end of a recurrence rule.  If no
         choice is indicated, the recurrence rule is considered
         to repeat forever.";
      case until {
        description
          "This case defines a way that limits the end of
           a recurrence rule in an inclusive manner.";
        leaf utc-until {
          type yang:date-and-time;
          description
            "This parameter specifies a date and time value to
             inclusively terminate the recurrence in UTC format.
             That is, if the value specified by this parameter is
             synchronized with the specified recurrence rule, it
             becomes the last instance of the recurrence rule.";
        }
      }
      case count {
        description
          "This case defines the number of occurrences at which
```

```
           to terminate the recurrence rule.";
        leaf count {
          type uint32 {
            range "1..max";
          }
          description
            "The positive number of occurrences at which to
             terminate the recurrence rule.";
        }
      }
    }
    uses recurrence-basic;
  }

  grouping recurrence-with-time-zone {
    description
      "A simple definition of recurrence to specify a recurrence
       rule with a time zone.";
    container recurrence-first {
      description
        "Specifies the first instance of the recurrence.  If
         unspecified, the recurrence is considered to start from
         the date and time when the recurrence pattern is first
         satisfied.";
      leaf start-time {
        type yang:date-and-time;
        description
          "Defines the date and time of the first instance
           in the recurrence set.";
      }
      leaf duration {
        type duration;
        description
          "When specified, it indicates how long the first
           occurrence lasts.  Unless specified otherwise, it also
           applies to all the other instances in the recurrence
           set.";
      }
    }
    leaf time-zone-identifier {
      type sys:timezone-name;
      description
        "Indicates the identifier for the time zone in a time
         zone database.  This parameter MUST be specified if either
         the 'start-time' or 'until' value is reported in local
         time format.  It MUST NOT be applied to date and time
         values that are specified in the format of UTC or time
         zone offset to UTC.";
    }
    choice recurrence-end {
      description
        "Modes to terminate the recurrence rule.  If no choice is
         indicated, the recurrence rule is considered to repeat
         forever.";
      case until {
        description
          "The end of the recurrence rule is indicated by a specific
           date-and-time value in an inclusive manner.";
```

```
        leaf until {
          type yang:date-and-time;
          description
            "Specifies a date and time value to inclusively terminate
             the recurrence.  That is, if the value specified by
             this parameter is synchronized with the specified
             recurrence, it becomes the last instance of the
             recurrence.";
        }
      }
      case count {
        description
          "The end of the recurrence is indicated by the number
           of occurrences.";
        leaf count {
          type uint32 {
            range "1..max";
          }
          description
            "The positive number of occurrences at which to
             terminate the recurrence.";
        }
      }
    }
    uses recurrence-basic;
  }

  grouping recurrence-utc-with-periods {
    description
      "This grouping defines an aggregate set of repeating
       occurrences with UTC time format.  The recurrence instances
       are specified by the occurrences defined by both the
       recurrence rule and 'period-timeticks' list.  Duplicate
       instances are ignored.";
    uses recurrence-utc;
    list period-timeticks {
      key "period-start";
      description
        "A list of periods with timeticks formats.";
      leaf period-start {
        type yang:timeticks;
        must "(not(derived-from(../../frequency,"
           + "'schedule:secondly')) or (current() < 100)) and "
           + "(not(derived-from(../../frequency,"
           + "'schedule:minutely')) or (current() < 6000)) and "
           + "(not(derived-from(../../frequency,'schedule:hourly'))"
           + " or (current() < 360000)) and "
           + "(not(derived-from(../../frequency,'schedule:daily'))"
           + " or (current() < 8640000)) and "
           + "(not(derived-from(../../frequency,'schedule:weekly'))"
           + " or (current() < 60480000)) and "
           + "(not(derived-from(../../frequency,"
           + "'schedule:monthly')) or (current() < 267840000)) and "
           + "(not(derived-from(../../frequency,'schedule:yearly'))"
           + " or (current() < 3162240000))" {
          error-message
            "The 'period-start' must not exceed the frequency
             interval.";
```

```
        }
        description
          "Start time of the schedule within one recurrence.

           Given that the value is in timeticks format
           (i.e., 1/100 of a second), the values in the must
           statement translate to 100 = 1 s (secondly),
           6000 = 60 s = 1 min (minutely), and so on for all
           instances in the must statement invariant.";
      }
      leaf period-end {
        type yang:timeticks;
        description
          "End time of the schedule within one recurrence.
           The period start MUST be no later than the period
           end.";
      }
    }
  }

  grouping recurrence-time-zone-with-periods {
    description
      "This grouping defines an aggregate set of repeating
       occurrences with local time format and time zone specified.
       The recurrence instances are specified by the occurrences
       defined by both the recurrence rule and 'period' list.
       Duplicate instances are ignored.";
    uses recurrence-with-time-zone;
    list period {
      key "period-start";
      description
        "A list of periods with date-and-time formats.";
      uses period-of-time;
    }
  }

  grouping icalendar-recurrence {
    description
      "This grouping specifies properties of a recurrence rule.";
    reference
      "RFC 5545: Internet Calendaring and Scheduling Core Object
                 Specification (iCalendar), Section 3.8.5";
    uses recurrence-time-zone-with-periods;
    leaf-list bysecond {
      type uint32 {
        range "0..60";
      }
      description
        "Specifies a list of seconds within a minute.";
    }
    leaf-list byminute {
      type uint32 {
        range "0..59";
      }
      description
        "Specifies a list of minutes within an hour.";
    }
    leaf-list byhour {
```

```
        type uint32 {
          range "0..23";
        }
        description
          "Specifies a list of hours of the day.";
      }
      list byday {
        key "weekday";
        description
          "Specifies a list of days of the week.";
        leaf-list direction {
          when "derived-from(../../frequency, 'schedule:monthly') or "
             + "(derived-from(../../frequency, 'schedule:yearly') "
             + " and not(../../byyearweek))";

          type int32 {
            range "-53..-1|1..53";
          }
          description
            "When specified, it indicates the nth occurrence of a
             specific day within the monthly or yearly recurrence
             rule.  For example, within a monthly rule, +1 monday
             represents the first Monday within the month, whereas
             -1 monday represents the last Monday of the month.";
        }
        leaf weekday {
          type schedule:weekday;
          description
            "Corresponds to seven days of the week.";
        }
      }
      leaf-list bymonthday {
        type int32 {
          range "-31..-1|1..31";
        }
        description
          "Specifies a list of days of the month.";
      }
      leaf-list byyearday {
        type int32 {
          range "-366..-1|1..366";
        }
        description
          "Specifies a list of days of the year.";
      }
      leaf-list byyearweek {
        when "derived-from(../frequency, 'schedule:yearly')";
        type int32 {
          range "-53..-1|1..53";
        }
        description
          "Specifies a list of weeks of the year.";
      }
      leaf-list byyearmonth {
        type uint32 {
          range "1..12";
        }
        description
```

```
        "Specifies a list of months of the year.";
    }
    leaf-list bysetpos {
      type int32 {
        range "-366..-1|1..366";
      }
      description
        "Specifies a list of values that corresponds to the nth
         occurrence within the set of recurrence instances
         specified by the rule.  It must only be used in conjunction
         with another 'byxxx' (bysecond, byminute, etc.) rule
         part.";
    }
    leaf workweek-start {
      type schedule:weekday;
      description
        "Specifies the day on which the workweek starts.";
    }
    leaf-list exception-dates {
      type yang:date-and-time;
      description
        "Defines a list of exceptions for recurrence.";
    }
  }

  grouping schedule-status {
    description
      "This grouping defines common properties of scheduling
       status.";
    leaf state {
      type identityref {
        base schedule-state;
      }
      description
        "Indicates the current state of the schedule.";
    }
    leaf version {
      type uint16;
      description
        "Indicates the version number of the schedule.";
    }
    leaf schedule-type {
      type identityref {
        base schedule-type;
      }
      description
        "Indicates the schedule type.";
    }
    leaf local-time {
      type yang:date-and-time;
      config false;
      description
        "Reports the local time as used by the entity that
         hosts the schedule.";
    }
    leaf last-update {
      type yang:date-and-time;
      config false;
```

```
      description
        "Reports the timestamp of when the schedule is last
         updated.";
    }
    leaf counter {
      when "derived-from-or-self(../schedule-type, "
         + "'schedule:recurrence')";
      type yang:counter32;
      config false;
      description
        "The number of occurrences while invoking the scheduled
         action successfully.  The count wraps around when it reaches
         the maximum value.";
    }
    leaf last-occurrence {
      when "derived-from-or-self(../schedule-type, "
         + "'schedule:recurrence')";
      type yang:date-and-time;
      config false;
      description
        "Indicates the timestamp of last occurrence.";
    }
    leaf upcoming-occurrence {
      when "derived-from-or-self(../schedule-type, "
         + "'schedule:recurrence')"
         + "and derived-from-or-self(../state, 'schedule:enabled')";
      type yang:date-and-time;
      config false;
      description
        "Indicates the timestamp of next occurrence.";
    }
    leaf last-failed-occurrence {
      when "derived-from-or-self(../schedule-type, "
         + "'schedule:recurrence')";
      type yang:date-and-time;
      config false;
      description
        "Indicates the timestamp of last failed action triggered by
         the schedule.";
    }
    leaf failure-counter {
      when "derived-from-or-self(../schedule-type, "
         + "'schedule:recurrence')";
      type yang:counter32;
      config false;
      description
        "Counts the number of failures while invoking the scheduled
         action.";
    }
  }

  grouping schedule-status-with-time-zone {
    description
      "This grouping defines common properties of scheduling
       status, including timezone.";
    leaf time-zone-identifier {
      type sys:timezone-name;
      config false;
```

```
      description
        "Indicates the identifier for the time zone in a time
         zone database.";
    }
    uses schedule-status;
  }

  grouping schedule-status-with-name {
    description
      "This grouping defines common properties of scheduling
       status, including a schedule name.";
    leaf schedule-name {
      type string;
      description
        "The schedule identifier that uniquely identifies a
         schedule within a device, controller, network, etc.
         The unicity scope depends on the implementation.";
    }
    uses schedule-status;
  }
}

<CODE ENDS>
```

7. Security Considerations

This section is modeled after the template described in Section 3.7 of [YANG-GUIDE].

The "ietf-schedule" YANG module defines a data model that is designed to be accessed via YANG-based management protocols, such as NETCONF [RFC6241] and RESTCONF [RFC8040]. These YANG-based management protocols (1) have to use a secure transport layer (e.g., SSH [RFC4252], TLS [RFC8446], and QUIC [RFC9000]) and (2) have to use mutual authentication.

The Network Configuration Access Control Model (NACM) [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

The "ietf-schedule" module defines a set of identities, types, and groupings. These nodes are intended to be reused by other YANG modules. The module by itself does not expose any data nodes that are writable, data nodes that contain read-only state, or RPCs. As such, there are no additional security issues related to the "ietf-schedule" module that need to be considered.

Modules that use the groupings that are defined in this document should identify the corresponding security considerations, for example:

- Scheduling depends on reliable and accurate time synchronization. Inaccurate date and time setting can lead to scheduling events being triggered at incorrect intervals, potentially causing system failures or security vulnerabilities.
- Recurring events may conceal abnormal behavior or security threats, which may be drowned out by normal events, especially when they are triggered frequently.

- The absence of detailed logs and audit records of each occurrence trigger time and action results and therefore may make security incidents difficult to trace.
- Care must be taken when defining recurrences occurring very often and frequent that can be an additional source of attacks by keeping the system permanently busy with the management of scheduling.

8. IANA Considerations

8.1. The IETF XML Registry

This document has registered the following URI in the "IETF XML Registry" [RFC3688].

URI: urn:ietf:params:xml:ns:yang:ietf-schedule

Registrant Contact: The IESG.

XML: N/A; the requested URI is an XML namespace.

8.2. The YANG Module Names Registry

This document has registered the following YANG module in the "YANG Module Names" registry [RFC6020].

Name: ietf-schedule

Maintained by IANA: N

Namespace: urn:ietf:params:xml:ns:yang:ietf-schedule

Prefix: schedule

Reference: RFC 9922

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3231] Levi, D. and J. Schoenwaelder, "Definitions of Managed Objects for Scheduling Management Operations", RFC 3231, DOI 10.17487/RFC3231, January 2002, <https://www.rfc-editor.org/info/rfc3231>.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <https://www.rfc-editor.org/info/rfc3688>.

[RFC5545] Desruisseaux, B., Ed., "Internet Calendaring and Scheduling Core Object Specification (iCalendar)", RFC 5545, DOI 10.17487/RFC5545, September 2009, <https://www.rfc-editor.org/info/rfc5545>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <https://www.rfc-editor.org/info/rfc6020>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/info/rfc6991>.

[RFC7317] Bierman, A. and M. Bjorklund, "A YANG Data Model for System Management", RFC 7317, DOI 10.17487/RFC7317, August 2014, <https://www.rfc-editor.org/info/rfc7317>.

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/info/rfc7950>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <https://www.rfc-editor.org/info/rfc8341>.

[W3C.XML1.0] Bray, T., Ed., Paoli, J., Ed., Sperberg-McQueen, C. M., Ed., Maler, E., Ed., and F. Yergeau, Ed., "Extensible Markup Language (XML) 1.0 (Fifth Edition)", W3C Recommendation, 26 November 2008, <https://www.w3.org/TR/2008/REC-xml-20081126/>.

9.2. Informative References

[NETMOD-ECA-POLICY] Wu, Q., Bryskin, I., Birkholz, H., Liu, X., and B. Claise, "A YANG Data model for ECA Policy Management", Work in Progress, Internet-Draft, draft-ietf-netmod-eca-policy-01, 19 February 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-netmod-eca-policy-01>.

[NTPv5] Lichvar, M. and T. Mizrahi, "Network Time Protocol Version 5", Work in Progress, Internet-Draft, draft-ietf-ntp-ntpv5-07, 16 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-ntp-ntpv5-07>.

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

[RFC4252] Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Authentication Protocol", RFC 4252, DOI 10.17487/RFC4252, January 2006, <https://www.rfc-editor.org/info/rfc4252>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC7951] Lhotka, L., "JSON Encoding of Data Modeled with YANG", RFC 7951, DOI 10.17487/RFC7951, August 2016, <https://www.rfc-editor.org/info/rfc7951>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <https://www.rfc-editor.org/info/rfc8340>.

[RFC8413] Zhuang, Y., Wu, Q., Chen, H., and A. Farrel, "Framework for Scheduled Use of Resources", RFC 8413, DOI 10.17487/RFC8413, July 2018, <https://www.rfc-editor.org/info/rfc8413>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <https://www.rfc-editor.org/info/rfc9000>.

[RFC9657] Birrane, III, E., Kuhn, N., Qu, Y., Taylor, R., and L. Zhang, "Time-Variant Routing (TVR) Use Cases", RFC 9657, DOI 10.17487/RFC9657, October 2024, <https://www.rfc-editor.org/info/rfc9657>.

[YANG-CONFIG-SCHEDULE] Liu, X., Bryskin, I., Beeram, V. P., Saad, T., Shah, H. C., and O. G. de Dios, "A YANG Data Model for Configuration Scheduling", Work in Progress, Internet-Draft, draft-liu-netmod-yang-schedule-05, 1 March 2018, <https://datatracker.ietf.org/doc/html/draft-liu-netmod-yang-schedule-05>.

[YANG-GUIDE] Bierman, A., Boucadair, M., and Q. Wu, "Guidelines for Authors and Reviewers of Documents Containing YANG Data Models", Work in Progress, Internet-Draft, draft-ietf-netmod-rfc8407bis-28, 5 June 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-netmod-rfc8407bis-28>.

[YANG-NAC] Ma, Q., Wu, Q., Boucadair, M., and D. King, "A YANG Data Model and RADIUS Extension for Policy-Based Network Access Control", Work in Progress, Internet-Draft, draft-ietf-opsawg-ucl-acl-12, 3 February 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-opsawg-ucl-acl-12>.

[YANG-OAM] Contreras, L. M., Lopez, V., and Q. Wu, "A YANG Data Model for Network Diagnosis using Scheduled Sequences of OAM Tests", Work in Progress, Internet-Draft, draft-ietf-opsawg-scheduling-oam-tests-03, 20 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-opsawg-scheduling-oam-tests-03>.

[YANG-SCHEDULE] Qu, Y., Lindem, A., Kinzie, E., Fedyk, D., and M. Blanchet, "YANG Data Model for Scheduled Attributes", Work in Progress, Internet-Draft, draft-ietf-tvr-schedule-yang-08, 9 February 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-tvr-schedule-yang-08>.

Appendix A. Examples of Scheduling Format Representation

This section provides some examples to illustrate the use of the period and recurrence formats defined in Section 6. The following modules are used for illustration purposes and make examples verifiable:

```
module example-sch-usage-1 {
  yang-version 1.1;
  namespace "http://example.com/example-sch-usage-1";
  prefix "ex-schu-1";

  import ietf-schedule {
    prefix "schedule";
  }

  container generic-schedule-params {
    uses schedule:generic-schedule-params;
  }
  container schedule-status {
    uses schedule:schedule-status;
  }
}

module example-sch-usage-2 {
  yang-version 1.1;
  namespace "http://example.com/example-sch-usage-2";
  prefix "ex-schu2";

  import ietf-schedule {
    prefix "schedule";
  }

  container period-of-time {
    uses schedule:period-of-time;
  }
}

module example-sch-usage-3 {
  yang-version 1.1;
  namespace "http://example.com/example-sch-usage-3";
  prefix "ex-schu-3";

  import ietf-schedule {
```

```
    prefix "schedule";
  }

  container recurrence-basic {
    uses schedule:recurrence-basic {
      refine frequency {
        mandatory true;
      }
      refine interval {
        default 1;
      }
    }
  }
}

module example-sch-usage-4 {
  yang-version 1.1;
  namespace "http://example.com/example-sch-usage-4";
  prefix "ex-schu-4";

  import ietf-schedule {
    prefix "schedule";
  }

  container recurrence-utc {
    uses schedule:recurrence-utc;
  }
}

module example-sch-usage-5 {
  yang-version 1.1;
  namespace "http://example.com/example-sch-usage-5";
  prefix "ex-schu-5";

  import ietf-schedule {
    prefix "schedule";
  }

  container recurrence-with-time-zone {
    uses schedule:recurrence-with-time-zone;
  }
}

module example-sch-usage-6 {
  yang-version 1.1;
  namespace "http://example.com/example-sch-usage-6";
  prefix "ex-schu-6";

  import ietf-schedule {
    prefix "schedule";
  }

  container recurrence-utc-with-date-times {
    uses schedule:recurrence-utc-with-periods;
  }
}

module example-sch-usage-7 {
```

```
  yang-version 1.1;
  namespace "http://example.com/example-sch-usage-7";
  prefix "ex-schu-8";

  import ietf-schedule {
    prefix "schedule";

  container recurrence-time-zone-with-date-times {
    uses schedule:recurrence-time-zone-with-periods;
  }
}

module example-sch-usage-8 {
  yang-version 1.1;
  namespace "http://example.com/example-sch-usage-8";
  prefix "ex-schu-8";

  container icalendar-recurrence {
    uses schedule:icalendar-recurrence {
      refine workweek-start {
        default monday;
      }
    }
  }
}
```

For each example, only the message body is provided with JSON, which is used for encoding per the guidance in [RFC7951].

A.1. The "generic-schedule-params" Grouping

Figure 10 illustrates the example of a requested schedule that needs to start no earlier than 08:00 AM, January 1, 2025 and end no later than 8:00 PM, January 31, 2025 (Beijing time). Schedule requests that fail to meet the requirements are ignored by the system, as indicated by "discard-action".

```
{
  "example-sch-usage-1:generic-schedule-params": {
    "time-zone-identifier": "China/Beijing",
    "min-allowed-start": "2025-01-01T08:00:00",
    "max-allowed-end": "2025-01-31T20:00:00",
    "discard-action": "ietf-schedule:silently-discard"
  }
}
```

Figure 10: Generic Parameters with 'max-allowed-end' for Schedule Validation

To illustrate the difference between "max-allowed-end" and "validity" parameters, Figure 11 shows the example of a requested schedule that needs to start no earlier than 08:00 AM, January 1, 2025 (Beijing time). Schedule requests that fail to meet the requirements are ignored by the

system, as indicated by "discard-action". The requested schedule may end after 8:00 PM, January 31, 2025, but any occurrences that are generated after that time would not be considered as valid.

```
{
  "example-sch-usage-1:generic-schedule-params": {
    "time-zone-identifier": "China/Beijing",
    "validity": "2025-01-31T20:00:00",
    "min-allowed-start": "2025-01-01T08:00:00",
    "discard-action": "ietf-schedule:silently-discard"
  }
}
```

Figure 11: Generic Parameters with 'validity' for Schedule Validation

A.2. The "period-of-time" Grouping

Figure 12 shows an example of a period that starts at 08:00:00 UTC on January 1, 2025 and ends at 18:00:00 UTC on December 31, 2027.

```
{
  "example-sch-usage-2:period-of-time": {
    "period-start": "2025-01-01T08:00:00Z",
    "period-end": "2027-12-31T18:00:00Z"
  }
}
```

Figure 12: Simple Start/End Schedule

An example of a period that starts at 08:00:00 UTC on January 1, 2025 and lasts 15 days and 5 hours and 20 minutes is encoded as shown in Figure 13.

```
{
  "example-sch-usage-2:period-of-time": {
    "period-start": "2025-01-01T08:00:00Z",
    "duration": "P15DT05:20:00"
  }
}
```

Figure 13: Simple Schedule with Duration

An example of a period that starts at 2:00 AM in Los Angeles on November 19, 2025 and lasts 20 weeks is depicted in Figure 14.

```
{
  "example-sch-usage-2:period-of-time": {
    "period-start": "2025-11-19T02:00:00",
    "time-zone-identifier": "America/Los_Angeles",
    "duration": "P20W"
  }
}
```

Figure 14: Simple Schedule with Time Zone Indication

A.3. The "recurrence-basic" Grouping

Figure 15 indicates a recurrence of every 2 days, which starts immediately and repeats forever:

```
{
  "example-sch-usage-3:recurrence-basic": {
    "recurrence-description": "forever recurrence rule",
    "frequency": "ietf-schedule:daily",
    "interval": 2
  }
}
```

Figure 15: Simple Schedule with Recurrence

A.4. The "recurrence-utc" Grouping

Figure 16 indicates a recurrence from 8:00 AM to 9:00 AM every day, from December 1 to December 31, 2025 in UTC:

```
{
  "example-sch-usage-4:recurrence-utc": {
    "recurrence-first": {
      "start-time-utc": "2025-12-01T08:00:00Z",
      "duration": 3600
    },
    "frequency": "ietf-schedule:daily",
    "interval": 1,
    "utc-until": "2025-12-31T23:59:59Z"
  }
}
```

Figure 16: Simple Schedule with Recurrence in UTC

A.5. The "recurrence-with-time-zone" Grouping

Figure 17 indicates a recurrence of every 2 hours for 10 occurrences that lasts 10 minutes and starts at 3 PM on December 1, 2025 in New York:

```
{
  "example-sch-usage-5:recurrence-with-time-zone": {
    "recurrence-first": {
      "start-time": "2025-12-01T15:00:00",
      "duration": "PT00:10:00",
      "time-zone-identifier": "America/New_York"
    },
    "frequency": "ietf-schedule:hourly",
    "interval": 2,
    "count": 10
  }
}
```

Figure 17: Simple Schedule with Recurrence with Time Zone Indication

A.6. The "recurrence-utc-with-periods" Grouping

Figure 18 indicates a recurrence that occurs every two days starting at 9:00 AM and 3:00 PM for a duration of 30 minutes and 40 minutes, respectively, from 2025-06-01 to 2025-06-30 in UTC:

```
{
  "example-sch-usage-6:recurrence-utc-with-periods": {
    "recurrence-first": {
      "start-time-utc": "2025-06-01T09:00:00Z"
    },
    "frequency": "ietf-schedule:daily",
    "interval": 2,
    "utc-until": "2025-06-30T23:59:59Z",
    "period-timeticks": [
      {
        "period-start": "3240000",
        "period-end": "3420000"
      },
      {
        "period-start": "5400000",
        "period-end": "5640000"
      }
    ]
  }
}
```

Figure 18: Example of Recurrence With Date Times

A.7. The "recurrence-time-zone-with-periods" Grouping

Figure 19 indicates a recurrence that occurs every 30 minutes and lasts for 15 minutes from 9:00 AM to 5:00 PM and two extra occurrences at 6:00 PM and 6:30 PM with each lasting for 20 minutes on 2025-12-01 (New York):

```
{
  "example-sch-usage-7:recurrence-time-zone-with-periods": {
    "recurrence-first": {
      "start-time": "2025-12-01T09:00:00",
      "duration": "PT00:15:00",
      "time-zone-identifier": "America/New_York"
    },
    "frequency": "ietf-schedule:minutely",
    "interval": 30,
    "until": "2025-12-01T17:00:00Z",
    "period": [
      {
        "period-start": "2025-12-01T18:00:00",
        "duration": "PT00:20:00"
      },
      {
        "period-start": "2025-12-01T18:30:00",
        "duration": "PT00:20:00"
      }
    ]
  }
}
```

Figure 19: Example of Advanced Recurrence Schedule

A.8. The "icalendar-recurrence" Grouping

Figure 20 indicates 10 occurrences at 8:00 AM (EST) every last Saturday of the month starting in January 2024:

```
{
  "example-sch-usage-8:icalendar-recurrence": {
    "recurrence-first": {
      "start-time": "2024-01-27T08:00:00",
      "time-zone-identifier": "America/New_York"
    },
    "frequency": "ietf-schedule:monthly",
    "count": 10,
    "byday": [
      {
        "direction": [
          -1
        ],
        "weekday": "saturday"
      }
    ]
  }
}
```

Figure 20: Simple iCalendar Recurrence

Figure 21 is an example of a recurrence that occurs on the last workday of the month until December 25, 2025, starting January 1, 2025:

```
{
  "example-sch-usage-8:icalendar-recurrence": {
    "recurrence-first": {
      "start-time": "2025-01-01"
    },
    "frequency": "ietf-schedule:monthly",
    "until": "2025-12-25",
    "byday": [
      {
        "weekday": "monday"
      },
      {
        "weekday": "tuesday"
      },
      {
        "weekday": "wednesday"
      },
      {
        "weekday": "thursday"
      },
      {
        "weekday": "friday"
      }
    ],
    "bysetpos": [
      -1
    ]
  }
}
```

Figure 21: Example of Advanced iCalendar Recurrence

Figure 22 indicates a recurrence that occurs every 20 minutes from 9:00 AM to 4:40 PM (UTC), with the exclusion of the occurrence starting at 10:20 AM on 2025-12-01:

```
{
  "example-sch-usage-8:icalendar-recurrence": {
    "recurrence-first": {
      "start-time": "2025-12-01T09:00:00Z"
    },
    "until": "2025-12-01T16:40:00Z",
    "frequency": "ietf-schedule:minutely",
    "byminute": [
      0,
      20,
      40
    ],
    "byhour": [
      9,
      10,
      11,
      12,
      13,
      14,
      15,
      16
    ],
    "exception-dates": [
      "2025-12-01T10:20:00Z"
    ]
  }
}
```

Figure 22: Example of Advanced iCalendar Recurrence with Exceptions

A.9. The "schedule-status" Grouping

Figure 23 indicates the scheduled recurrence status of Figure 22 at the time of 12:15 PM on 2025-12-01 (UTC):

```
{
  "example-sch-usage-1:schedule-status": {
    "state": "ietf-schedule:enabled",
    "version": 1,
    "schedule-type": "ietf-schedule:recurrence",
    "counter": 9,
    "last-occurrence": [
      "2025-12-01T12:00:00Z"
    ],
    "upcoming-occurrence": [
      "2025-12-01T12:20:00Z"
    ]
  }
}
```

Figure 23: Example of a Schedule Status

At the time of 12:15 PM on 2025-12-01 (UTC), the recurring event occurred at (note that the occurrence at 10:20 AM is excluded): 9:00, 9:20, 9:40, 10:00, 10:40, 11:00, 11:20, 11:40, and 12:00. The last occurrence was at 12:00, and the upcoming one is at 12:20.

Appendix B. Examples of Using/Extending the "ietf-schedule" Module

This non-normative section shows two examples for how the "ietf-schedule" module can be used or extended for scheduled events or attributes based on date and time.

B.1. Example: Schedule Tasks to Execute Based on a Recurrence Rule

Scheduled tasks can be used to execute specific actions based on certain recurrence rules (e.g., every Friday at 8:00 AM). The following example module, which "uses" the "icalendar-recurrence" grouping from the "ietf-schedule" module, shows how a scheduled task could be defined with different features used for options.

```
module example-scheduled-backup {
  yang-version 1.1;
  namespace "http://example.com/example-scheduled-backup";
  prefix "ex-scback";

  import ietf-inet-types {
    prefix "inet";
  }

  import ietf-schedule {
    prefix "schedule";
  }

  organization
    "Example, Inc.";

  contact
    "Support at example.com";

  description
    "Example of a module defining a scheduled-based backup
     operation.";

  revision "2023-01-19" {
    description
      "Initial version.";
    reference
      "RFC 9922: A YANG Data Model for Scheduling.";
    }

  container scheduled-backup-tasks {
    description
      "A container for backing up all current running configurations
       on the device.";
    list tasks {
      key "task-id";
```

```
      description
        "The list of backing up tasks on this device.";
      leaf task-id {
        type string;
        description
          "The task identifier that uniquely identifies a scheduled
           backup task.";
      }
      choice local-or-remote {
        description
          "Specifies whether the configuration to be backed up is
           local or remote.";
        case local {
          description
            "Configuration parameters for the backing up of local
             devices.";
          leaf local {
            type empty;
            description
              "The parameter specifies the configuration to be
               backed up is on the local device.";
          }
        }
        case remote {
          description
            "Configuration parameters for backing up of remote
             devices.";
          leaf remote {
            type inet:domain-name;
            description
              "The parameter specifies the remote device domain
               name.";
          }
        }
      }

      container basic-recurrence-schedules {
        if-feature schedule:basic-recurrence;
        description
          "Basic recurrence schedule specification, which only
           applies when the schedule:basic-recurrence feature
           is supported.";
        leaf schedule-id {
          type string;
          description
            "The schedule identifier for this recurrence rule.";
        }
        uses schedule:recurrence-basic {
          refine frequency {
            mandatory true;
          }
          refine interval {
            default 1;
          }
        }
      }

      container icalendar-recurrence-schedules {
```

```
        if-feature schedule:icalendar-recurrence;
        description
          "Basic recurrence schedule specification, which only
           applies when the schedule:icalendar-recurrence feature
           is supported.";
        leaf schedule-id {
          type string;
          description
            "The schedule identifier for this recurrence rule.";
        }
        uses schedule:icalendar-recurrence {
          refine workweek-start {
            default monday;
          }
        }
      }
    }

    list schedule-set {
      key "schedule-name";
      description
        "Schedule status list for the backup tasks.";
      uses schedule:schedule-status-with-name;
    }
  }
}
```

B.2. Example: Schedule Network Properties to Change Based on Date and Time

Network properties may change over a specific period of time or based on a recurrence rule, e.g., [RFC9657]. The following example module, which augments the "recurrence-utc-with-periods" grouping from the "ietf-schedule" module, shows how a scheduled attribute could be defined.

```
module example-scheduled-link-bandwidth {
  yang-version 1.1;
  namespace "http://example.com/example-scheduled-link-bandwidth";
  prefix "ex-scattr";

  import ietf-network {
    prefix "nw";
    reference
      "RFC 8345: A YANG Data Model for Network Topologies";
  }

  import ietf-schedule {
    prefix "schedule";
    reference
      "RFC 9922: A YANG Data Model for Scheduling";
  }

  organization
    "Example, Inc.";

  contact
```

```
    "Support at example.com";

  description
    "Example of a module defining a scheduled link bandwidth.";

  revision "2023-01-19" {
    description
      "Initial version.";
    reference
      "RFC 9922: A YANG Data Model for Scheduling";
    }

  grouping link-bandwidth-grouping {
    description
      "Grouping of the link bandwidth definition.";
    leaf scheduled-bandwidth {
      type uint64;
      units "Kbps";
      description
        "Bandwidth values, expressed in kilobits per second.";
    }
  }

  container link-attributes {
    description
      "Definition of link attributes.";
    list link {
      key "source-node destination-node";
      description
        "Definition of link attributes.";
      leaf source-node {
        type nw:node-id;
        description
          "Indicates the source node identifier.";
      }
      leaf destination-node {
        type nw:node-id;
        description
          "Indicates the source node identifier.";
      }

      leaf default-bandwidth {
        type uint64;
        units "Kbps";
        description
          "Bandwidth value used for periods that don't match
           a schedule.";
      }

      choice time-variant-type {
        description
          "Controls the schedule type.";
        case period {
          uses schedule:period-of-time;
        }
        case recurrence {
          uses schedule:recurrence-utc-with-periods {
            augment "period-timeticks" {
```

```
              description
                "Specifies the attributes inside each
                 'period-timeticks' entry.";
              uses link-bandwidth-grouping;
            }
          }
        }
      }
    }
  }
}
```

Figure 24 shows a configuration example in XML [W3C.XML1.0] of a link's bandwidth that is scheduled between 2025-12-01 0:00 UTC to the end of 2025-12-31 with a daily schedule. In each day, the bandwidth value is scheduled to be 500 Kbps between 1:00 AM to 6:00 AM and 800 Kbps between 10:00 PM to 11:00 PM. The bandwidth value that is not covered by the period above is 1000 Kbps.

```
<?xml version="1.0" encoding="utf-8"?>
<link-attributes
  xmlns="http://example.com/example-scheduled-link-bandwidth"
  xmlns:schedule="urn:ietf:params:xml:ns:yang:ietf-schedule">
  <link>
    <source-node>ne1</source-node>
    <destination-node>ne2</destination-node>
    <default-bandwidth>1000</default-bandwidth>
    <recurrence-first>
      <utc-start-time>2025-12-01T01:00:00Z</utc-start-time>
    </recurrence-first>
    <frequency>schedule:daily</frequency>
    <utc-until>2025-12-31T23:59:59Z</utc-until>
    <period-timeticks>
      <period-start>360000</period-start>
      <period-end>2160000</period-end>
      <scheduled-bandwidth>500</scheduled-bandwidth>
    </period-timeticks>
    <period-timeticks>
      <period-start>7920000</period-start>
      <period-end>8280000</period-end>
      <scheduled-bandwidth>800</scheduled-bandwidth>
    </period-timeticks>
  </link>
</link-attributes>
```

Figure 24: Example of Scheduled Link's Bandwidth

Appendix C. Examples of Using the "ietf-schedule" Module for Scheduled Use of Resources Framework

This section exemplifies how the architecture for supporting scheduled reservation of Traffic Engineering (TE) resources in [RFC8413] might leverage the "period-of-time" grouping defined in the "ietf-schedule" module to implement scheduled use of resources.

The following example module shows how a scheduled link capacity reservation could be defined.

```
module example-sch-capacity-res {
  yang-version 1.1;
  namespace "http://example.com/example-sch-capacity-res";
  prefix "ex-schecaparev";

  import ietf-network-topology {
    prefix "nt";
  }

  import ietf-schedule {
    prefix "schedule";
  }

  container link-capability-reservations {
    list scheduled-link-capacity {
      key "schedule-id";
      leaf schedule-id {
        type string;
      }
      leaf link-id {
        type nt:link-id;
      }
      leaf reserved-capability {
        type uint64;
        units "Mbps";
      }
      uses schedule:period-of-time;
    }
  }
}
```

Section 4 of [RFC8413] defines the reference architecture for scheduled use of resources. The service requester sends a request to a Path Computation Element (PCE) and includes the parameters of the Label Switched Path (LSP) that the requester wishes to supply. The configuration example to provide the scheduled resource is shown in Figure 25.

```
<?xml version="1.0" encoding="utf-8"?>
<link-capability-reservations
  xmlns="http://example.com/example-sch-capacity-res"
  xmlns:schedule="urn:ietf:params:xml:ns:yang:ietf-schedule">
   <scheduled-link-capacity>
      <schedule-id>1</schedule-id>
      <link-id>1-2-1</link-id>
      <reserved-capability>500</reserved-capability>
      <period-start>2025-03-10T08:00:00Z</period-start>
      <period-end>2025-03-10T09:00:00Z</period-end>
   </scheduled-link-capacity>
   <scheduled-link-capacity>
      <schedule-id>2</schedule-id>
      <link-id>2-1-1</link-id>
      <reserved-capability>400</reserved-capability>
      <period-start>2025-04-01T00:00:00Z</period-start>
      <duration>PT09:00:00</duration>
   </scheduled-link-capacity>
   <scheduled-link-capacity>
      <schedule-id>3</schedule-id>
      <link-id>2-1-1</link-id>
      <reserved-capability>500</reserved-capability>
      <period-start>2025-04-01T09:00:00Z</period-start>
      <period-end>2025-04-01T23:59:59Z</period-end>
   </scheduled-link-capacity>
</link-capability-reservations>
```

Figure 25: Example of Scheduled Link's Bandwidth Reservation

Acknowledgments

This work is derived from [YANG-NAC]. There is a desire from the OPSAWG to see this module separately defined for wide use in scheduling context.

Thanks to Adrian Farrel, Wei Pan, Tianran Zhou, Joe Clarke, Steve Baillargeon, Dhruv Dhody, Robert Wilton, and Italo Busi for their valuable comments and inputs to this work.

Many thanks to the authors of [YANG-SCHEDULE], [YANG-OAM], and [NETMOD-ECA-POLICY] for the constructive discussion during IETF#118.

Other related efforts were explored in the past, e.g., [YANG-CONFIG-SCHEDULE].

Thanks to Reshad Rahman for the great YANG Doctors review, Mahesh Jethanandani for the AD review, Per Andersson for the OPSDIR review, Peter Yee for the GENART review, and Acee Lindem for the RTGDIR review.

Thanks to Éric Vyncke, Erik Kline, and Mike Bishop for the IESG review.

Authors' Addresses

Qiufang Ma (EDITOR)
Huawei
101 Software Avenue, Yuhua District
Jiangsu
210012
China
Email: maqiufang1@huawei.com

Qin Wu
Huawei
101 Software Avenue, Yuhua District
Jiangsu
210012
China
Email: bill.wu@huawei.com

Mohamed Boucadair (EDITOR)
Orange
35000 Rennes
France
Email: mohamed.boucadair@orange.com

Daniel King
Lancaster University
United Kingdom
Email: d.king@lancaster.ac.uk