Condition match modes

Note: This feature is currently in beta and is available only to early adopters.

Compliance condition match modes control how multiple input lines are matched against the validated source (backup/MCP output).

All Compliance Condition types support three match modes:

- **Loose**
- **Sequential**
- **Strict**

Loose mode verifies that all input lines are present in the source. Their order does not matter, and additional lines may appear between them.

Sequential mode verifies that all input lines are present in the source in the same order as entered. Additional lines may appear between them.

Strict mode verifies that all input lines are present in the source in the same order as entered and consecutively. No additional lines may appear between the matching lines.

Match mode	Line order matters	Lines must be consecutive
Loose	No	No
Sequential	Yes	No
Strict	Yes	Yes

Reference source:

The compliance condition examples throughout this article are evaluated against the following configuration backup. This is a theoretical example containing only the relevant configuration lines; many config lines that would normally appear in a real backup are omitted for brevity.

```
hostname R1
!
interface Ethernet1
 description Uplink
 ip address 10.0.0.1 255.255.255.0
 no shutdown
!
interface Ethernet2
 description LAN
 ip address 192.168.1.1 255.255.255.0
 no shutdown
!
interface Loopback0
 description Management Loopback
 ip address 10.255.255.1 255.255.255.255
!
router ospf
 router-id 1.1.1.1
 network 10.0.0.0 0.0.0.255 area 0
!
```

Note: The Compliance module can validate MCP automation outputs in addition to device configurations. To validate an automation output, select *Source Mass Config Push Result* on the Compliance preset screen. The same condition types and matching principles described in this article apply when evaluating automation output.

Loose mode:

Source contains

The following examples demonstrate how Loose mode evaluates input lines when using the *Source contains* condition type.

Condition 1:

```
description LAN
hostname R1
```

Status: Condition successful

Although the order of the input lines differs from the source, both lines are present.

Condition 2:

```
hostname R1
 description WAN
```

Status: Condition failed

The line *description WAN* does not exist in the source.

Source matches regex

The following examples demonstrate how Loose mode evaluates input lines when using the *Source matches regex* condition type.

Condition 3:

```
router ospf
hostname R[0-9]+
```

Status: Condition successful

This verifies that the device follows the *R<number>* hostname convention and has OSPF enabled. The order of the matches does not matter.

Condition 4:

```
router ospf
hostname R[A-Z]+
```

Status: Condition failed

The *router ospf* line matches the source, but *hostname R[A-Z]+* regex does not match *hostname R1*. Because all input lines must match the source, the condition fails.

Sequential mode:

Source contains

The following examples demonstrate how Sequential mode evaluates input lines when using the *Source contains* condition type.

Condition 5:

```
interface Ethernet1
 ip address 10.0.0.1 255.255.255.0
```

Status: Condition successful

Both input lines appear in the source in the correct order. The "description Uplink" line between them is ignored.

Condition 6:

```
ip address 10.0.0.1 255.255.255.0
interface Ethernet1
```

Status: Condition failed

The input lines appear in the source in reverse order.

Source matches regex

The following examples demonstrate how Sequential mode evaluates input lines when using the *Source matches regex* condition type.

Note: The regular expressions used to match IP addresses in the examples are intentionally simplified for clarity. They are therefore more permissive than a full IPv4 validation pattern and may also match invalid IP addresses.

Condition 7:

```
router ospf
 network [0-9]+\.[0-9]+\.[0-9]+\.[0-9]+ 0\.0\.0\.[0-9]+ area [0-9]+
```

Status: Condition successful

This verifies that an OSPF network statement appears after the "router ospf" command. Other matching lines may appear between the two lines.

Condition 8:

```
network [0-9]+\.[0-9]+\.[0-9]+\.[0-9]+ 0\.0\.0\.[0-9]+ area [0-9]+
router ospf
```

Status: Condition failed

Both lines are present in the source, but they appear in the opposite order. Sequential mode therefore fails.

Strict mode:

Source contains

The following examples demonstrate how Strict mode evaluates input lines when using the *Source contains* condition type.

Condition 9:

```
interface Ethernet1
 description Uplink
 ip address 10.0.0.1 255.255.255.0
```

Status: Condition successful

All three lines appear consecutively in the source.

Condition 10:

```
interface Ethernet1
 ip address 10.0.0.1 255.255.255.0
```

Status: Condition failed

The source contains the line "description Uplink" between the two input lines. Strict mode therefore fails.

Source matches regex

The following examples demonstrate how *Strict mode* evaluates input lines when using the *Source matches regex* condition type.

Condition 11:

```
interface Loopback[0-9]+
 description Management Loopback
 ip address 10\.255\.255\.[0-9]+ 255\.255\.255\.255
```

Status: Condition successful

This verifies that a Loopback interface with a variable interface number has the required "Management Loopback" description and uses a /32 address matching the 10.255.255.x pattern.

Condition 12:

```
interface Loopback[0-9]+
ip address 10\.255\.255\.[0-9]+ 255\.255\.255\.255
```

Status: Condition failed

Both lines match in the source, but the *description Management Loopback* line appears between them. Strict mode requires the matching lines to be consecutive, so the condition fails.

Common use cases:

The following examples illustrate common configuration validation scenarios and the match mode best suited to each.

Config to validate	Match mode	Rationale
Baseline configuration and security hardening - NTP servers, DNS servers, syslog host, password encryption, telnet disabled, HTTP disabled	Loose	The required lines are independent of each other and may appear in different parts of the configuration. Only their presence matters.
Firewall and ACL rules, route-maps	Sequential	The required lines must appear in a specific order, while other configuration lines may appear between them.

Multiline MOTD banner, interface configurations	Strict	The required lines form a complete configuration block and must appear consecutively without any other lines between them.