Explainable Network Verification via Subspecifications

User Study - Introduction of Background Knowledge

Background: Explainable Network Verification

Network verifiers often give **YES** **NO** (with a counterexample) answers, without explaining ***why***.

Example: suppose we want a BGP policy blocking the private prefix `192.168.0.0/16`.

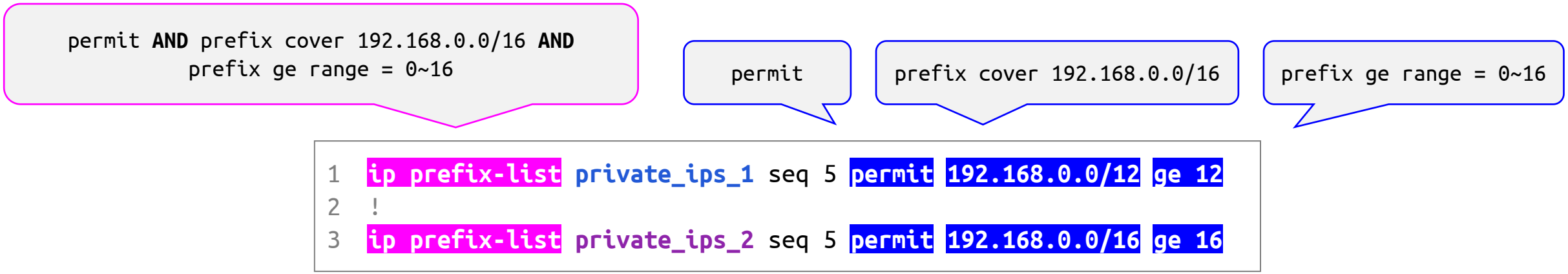
```
1  ip prefix-list private_ips_1 seq 5 permit 192.168.0.0/12 ge 12
2  !
3  ip prefix-list private_ips_2 seq 5 permit 192.168.0.0/16 ge 16
```

Both of them pass the verification.

However, **`private_ips_2`** is more ***precise*** than **`private_ips_1`**.
Overly restrictive filter may block more prefix than necessary.

Explainable Network Verification

Why a specific field, line, or block of the configuration satisfies the specification?

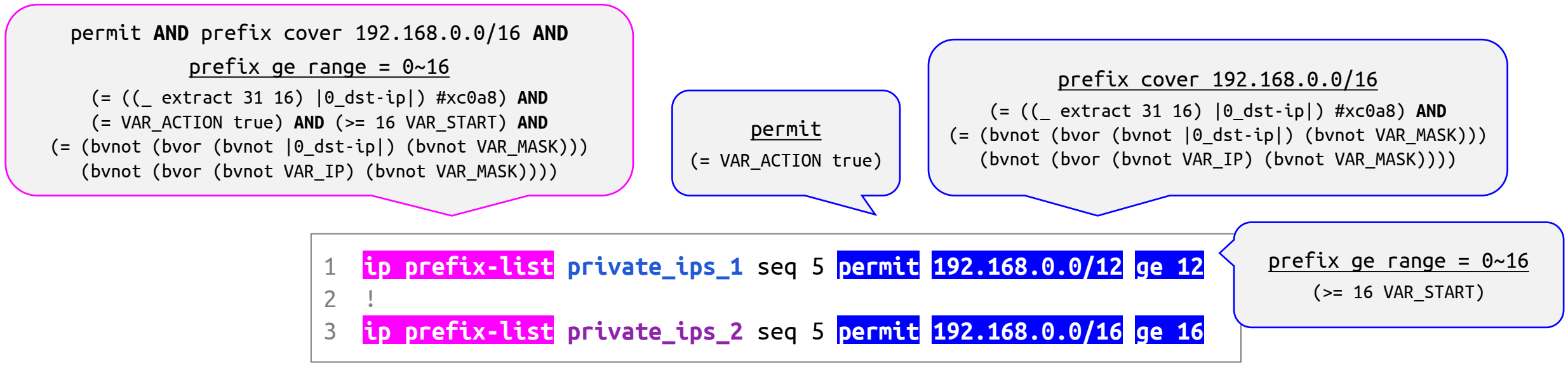


Both of them pass the verification.

`private_ips_2` is the more precise option.

Explainable Network Verification via Subspecifications

( ): of that field, line, or block, while preserving the prior verification success.



How to use subspec

Modification subspec: preserves the verification property.

prefix ge range = 0~16
(>= 16 VAR_START)

```
1  ip prefix-list private_ips_1 seq 5 permit 192.168.0.0/12 ge 18
```

Modification subspec: may violate the property.
(sound but not complete)

prefix ge range = 0~16
(>= 16 VAR_START)

```
1  ip prefix-list private_ips_1 seq 5 permit 192.168.0.0/12 ge 14
```

How to use subspec

[redacted] can modify to anything.

empty

```
1  ip prefix-list private_ips_1 seq 5 permit 192.168.0.0/12 ge 14
```

[redacted] can directly remove that line.

(or a line contains [redacted] and the subspec is [redacted])

empty

```
1  ip prefix-list private_ips_1 seq 5 permit 192.168.0.0/12 ge 12
```

Tips for User Study

1. In this user study, we consider two granularities: and subspecs.

2. In this user study, the eBGP route selection process only involves .

3. In this user study, the route-map naming rule is .

Explainable Network Verification via Subspecifications

User Study - Introduction of Background Knowledge

Thank you for participating in this user study!

Tips for Subspecs

1. modifications ***satisfying*** the subspec bounds are guaranteed to preserve the verified specifications

```
1  ip prefix-list private_ips_1 seq 5 permit 192.168.0.0/16 ge 16
```

2. modifications ***exceeding*** the subspec bounds the verified specifications ***(sound but not complete)***

```
1  ip prefix-list private_ips_1 seq 5 deny 192.168.0.0/17 ge 17
```

permit **AND** prefix cover 192.168.0.0/16 **AND**
prefix ge range = 0~16

```
(= ((_ extract 31 16) |0_dst-ip|) #xc0a8) AND
(= VAR_ACTION true) AND (>= 16 VAR_START) AND
(= (bvnot (bvor (bvnot |0_dst-ip|) (bvnot VAR_MASK)))
   (bvnot (bvor (bvnot VAR_IP) (bvnot VAR_MASK))))
```

permit

```
(= VAR_ACTION true)
```

prefix cover 192.168.0.0/16

```
(= ((_ extract 31 16) |0_dst-ip|) #xc0a8) AND
(= (bvnot (bvor (bvnot |0_dst-ip|) (bvnot VAR_MASK)))
   (bvnot (bvor (bvnot VAR_IP) (bvnot VAR_MASK))))
```

```
1  ip prefix-list private_ips_1 seq 5 permit 192.168.0.0/12 ge 12
```

prefix ge range = 0~16

```
(>= 16 VAR_START)
```

Tips for Empty Subspecs

1. ***safely modify*** that field with c without breaking the verified specifications

```
1  ip prefix-list private_ips_1 seq 5 deny 0.0.0.0/0 ge 0
```

2. ***safely remove*** that line with without breaking the verified specifications
   (or a line contains and the field-level subspec is)

```
1 ip prefix-list private_ips_1 seq 5 permit 192.168.0.0/12 ge 12
```