Transitive BGPsec is Difficult/Infeasible

https://datatracker.ietf.org/doc/draft-wang-sidrops-bgpsec-transition/

Yangfei Guo

Authors: Ke Xu, Xiaoliang Wang, Zhutao Liu, Qi Li, and Yangfei Guo

SIDROPS WG
IETF 124
Nov. 2025

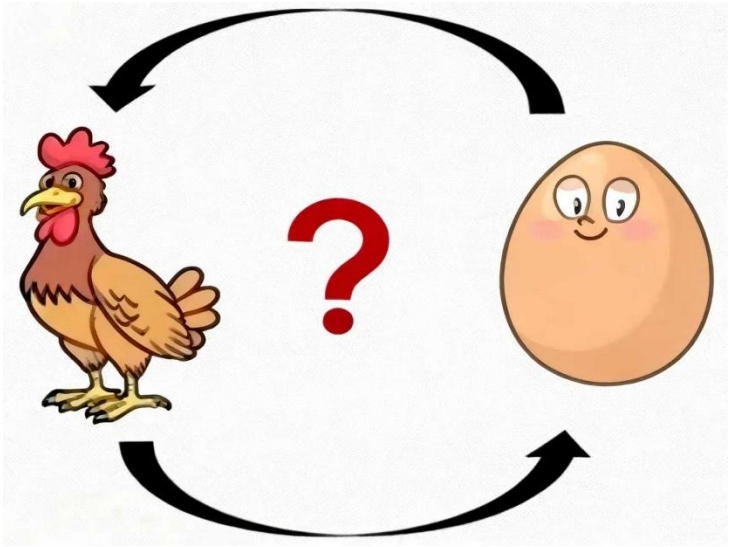
Motivation

- Traditional BGPsec is hard to achieve incremental deployment
  - Only two BGPsec public keys are available in RPKI as of Oct. 28, 2025.
    - https://console.rpki-client.org/rpki.json
- Barriers
  - OPEN Negotiation
    - BGPsec Capability
    - Multiprotocol Capability
  - UPDATE
    - non-transitive
- **Goal**: Render BGPsec transitive so as to facilitate the deployment of BGPsec.


```
'metadata": {
        "buildmachine": "dus-vrp-gen1.rpki-client.org",
        "buildtime": "2025-10-28T01:43:37Z",
        "elapsedtime": 326,
        "usertime": 489,
        "systemtime": 614,
        "ccr_mfts_hash": "0Q70tsriPn5EbuZYR7spM2hAo9evWmbHYLxJANNe8xI
        "ccr_vrps_hash": "LO1aBOJyfbOR5/PBuyXFs59mYVxCWGAfxvYDoxU2LjA
        "ccr_vaps_hash": "+/vpLWLNaA4aOqHKdO/5Nlhy7e9WDsHE3dhayNKcJTM
        "ccr_brks_hash": "ul+0Sc77a6APNhJ5YqLupuhn/oUSu92t6cbkuLwWwdI
        "ccr_tas_hash": "ubpmsrzVTkgSJJ9g7S3pNXZwzEj/hI8bw19ZhnA95x8=
        "roas": 341178,
        "failedroas": 14,
        "invalidroas": 0,
        "spls": 0,
        "failedspls": 0,
        "invalidspls": 0,
        "aspas": 134,
        "failedaspas": 0,
        "invalidaspas": 0,
        "bgpsec_pubkeys": 2,
        "certificates": 48930,
        "invalidcertificates": 14,
        "nonfunctionalcas": 139,
        "taks": 0,
        "tals": 5,
        "invalidtals": 0,
        "talfiles": [
                "/etc/rpki/afrinic.tal",
                "/etc/rpki/apnic.tal",
                "/etc/rpki/lacnic.tal",
                "/etc/rpki/ripe.tal",
                "/etc/rpki/arin.tal"
        ],
        "manifests": 48916,
        "failedmanifests": 139,
        "crls": 48777,
        "gbrs": 2,
        "repositories": 105,
        "vrps": 773125,
        "uniquevrps": 767554,
        "vsps": 0,
```

Motivation (Cont.)

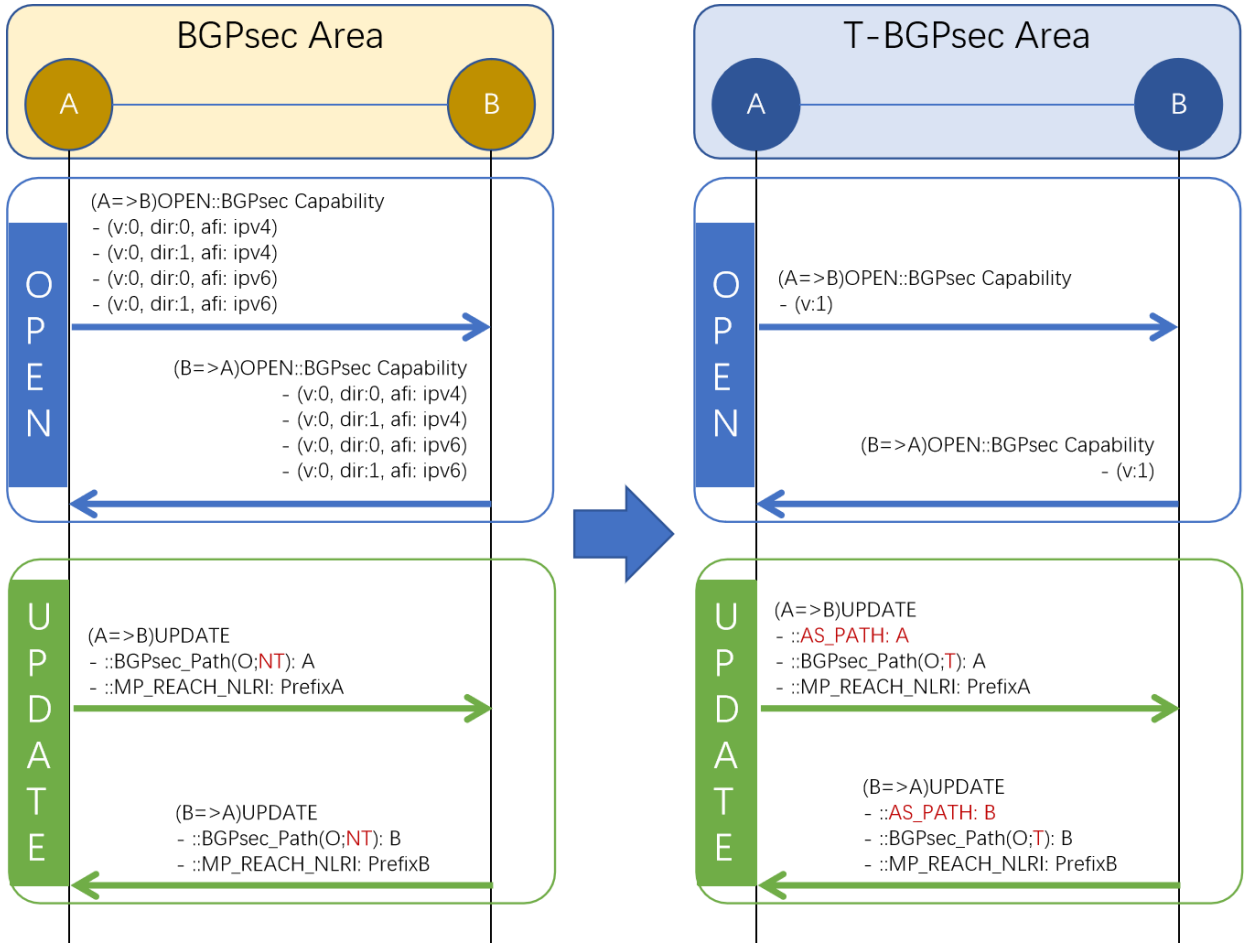
- Chicken-and-Egg Problem
  - The design of BGPsec prioritizes technology over deployer preference.
  - There are fewer incentives for early adopters in today's routing security mechanisms.[1]
  - If there are no benefits to the early adopters, why become early adopters in the first place?
- The New BGPsec Mechanism
  - Incremental deployability should allow early adopters to benefit, avoiding the chicken-and-egg problem.



[1] NTIA Supports FCC Internet Routing Security Proposal. https://www.ntia.gov/press-release/2024/ntia-supports-fcc-internet-routing-security-proposal.

Overview: Transitive BGPsec

- **Packet formats of BGPsec are reused**
- BGP Open::BGPsec Capability
  - Version is upgraded
- BGP Update
  - AS_PATH is reserved
  - BGPsec_PATH: non-transitive => transitive
- Signature choices
  - Onion-like signature (all ASes on the current path are involved in a signature)
  - Pathlet signature (only previous ASN, local ASN, next ASN are involved in a signature)
- Questions
  - Compatibility with Native BGPsec



NT for non-transitive, T for transitive

BGP Open Negotiation

- Open Negotiation:
  - Version: 0 (BGPsec)
    - Dir: 0 for receiving, 1 for sending
    - AFI: ipv4 or ipv6
    - 4 BGPsec capablities
      - <0, ipv4>, <1, ipv4>
      - <0, ipv6>, <1, ipv6>
  - Version: 1 (Transitive BGPsec)
    - no need for Dir and AFI
- Why do we still need negotiation?
  - Notify the peer that this BGP speaker supports Transitive BGPsec.
  - Distinguish Transitive BGPsec from traditional BGPsec for compatibility.

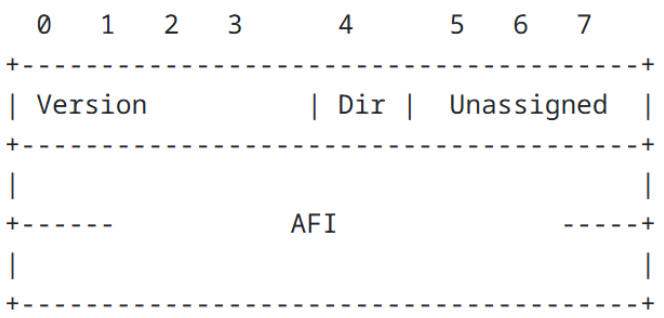


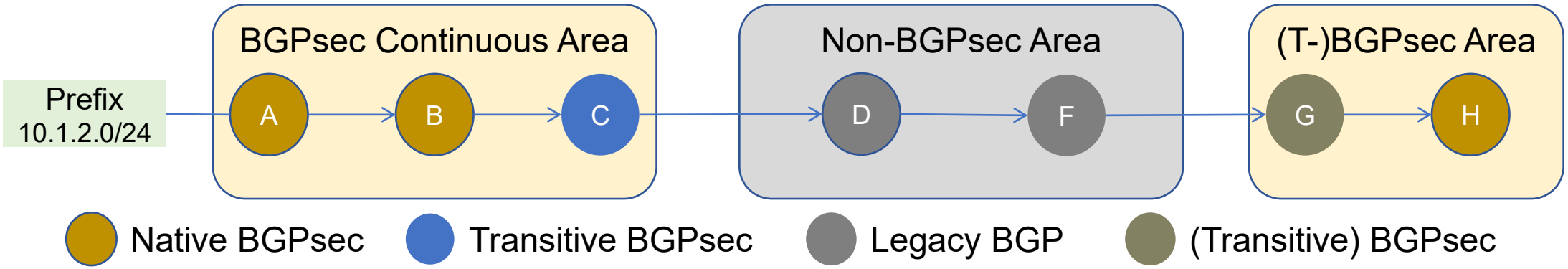
Figure 1: BGPsec Capability Format

In a BGPsec peering session, a peer is permitted to send UPDATE messages containing the BGPsec_PATH attribute if and only if:

o The given peer sent the BGPsec capability for a particular version of BGPsec and a particular address family with the Direction bit set to 1, and

o The other (receiving) peer sent the BGPsec capability for the same version of BGPsec and the same address family with the Direction bit set to 0.

BGP Update



1. non-transitive => transitive
2. AS_PATH attribute is reserved.
   - AS_PATH is mandatory attribute in legacy BGP.
   - AS_PATH is required for the purpose of transparently traversing legacy BGP.
3. Signature
   - Should we still use onion-like signature when BGPsec_PATH is broken?
   - We much prefer to use pathlet signature instead.
     - NOT the whole path BUT 3ASN: <previous ASN, local ASN, next ASN>.

Q1: Partial Path Signing

- If we want to facilitate smoother BGPsec deployment, we need to ease this requirement.
- The onion-like signature requires strong path integrity, while the pathlet signature requires only weak path integrity.
- Both types of signatures aim to achieve the same security targets.
- Given this, the pathlet signature is more applicable.

6.4. Partial Path Signing

"Partial path signing" means that a BGPsec AS can be permitted to sign an update that was received unsigned from a downstream neighbor. That is, the AS would add its ASN to the AS path and sign the (previously unsigned) update to other neighboring (upstream) BGPsec ASes.

6.4.1. Decision

It was decided that partial path signing in BGPsec will not be allowed. A BGPsec update must be fully signed, i.e., each AS in the AS path must sign the update. So, in a signed update, there must be a signature corresponding to each AS in the AS path.

6.4.2. Discussion

Partial path signing (as described above) implies that the AS path is not rigorously protected. Rigorous AS path protection is a key requirement of BGPsec [RFC7353]. Partial path signing clearly reintroduces the following attack vulnerability: if a BGPsec speaker is allowed to sign an unsigned update and if signed (i.e., partially or fully signed) updates would be preferred over unsigned updates, then a faulty, misconfigured, or subverted BGPsec speaker can manufacture any unsigned update it wants (by inserting a valid origin AS) and add a signature to it to increase the chance that its update will be preferred.

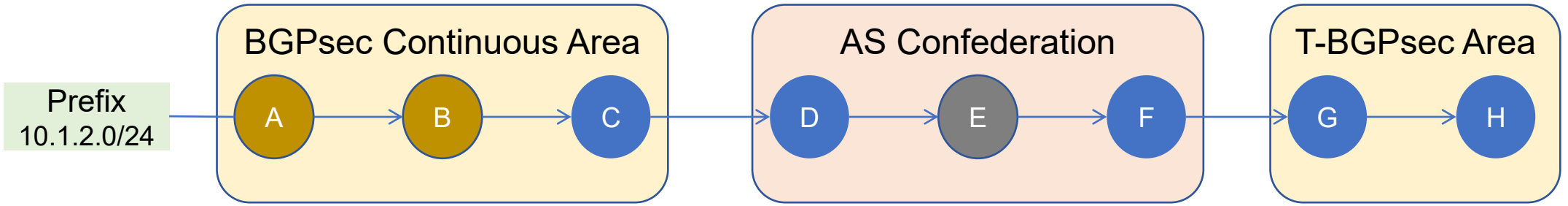
RFC 8374 - BGPsec Design Choices and Summary of Supporting Discussions

Q2: Copy-and-Paste Attack

- Once the previous ASN has been signed, it is impossible for an attacker to execute a copy-and-paste attack.
- If an attacker attempts to carry out such an attack, it needs to remove at least one signed segment. However, removing any segments would result in the update message being flagged as invalid.

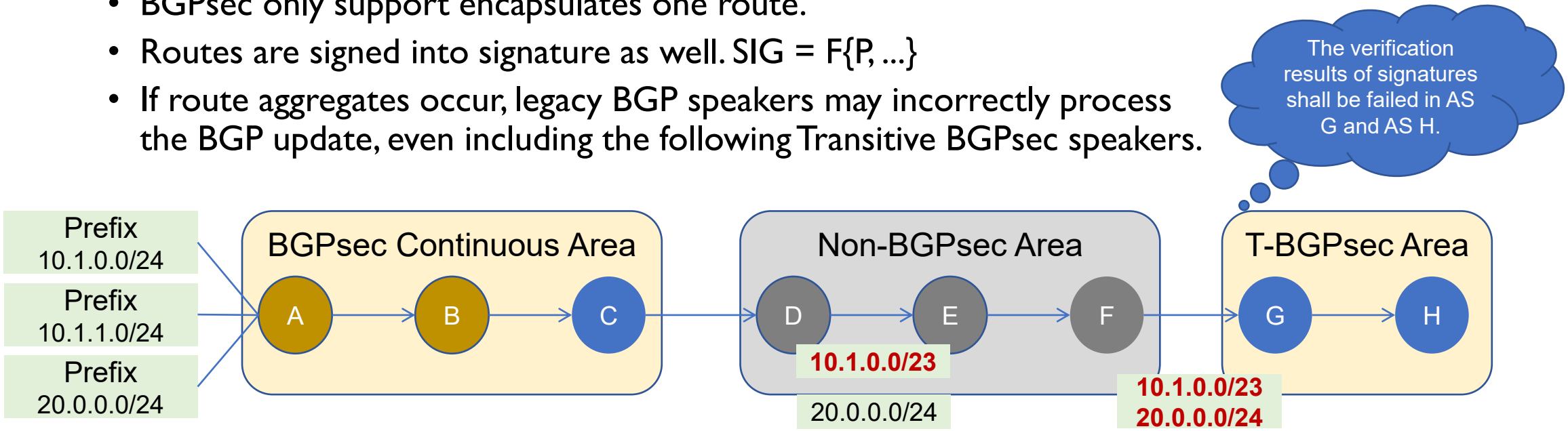
Q3: ASPP, RS, AS Confederation

- AS Path Prepending (ASPP) and Route Server (RS)
  - AS_PATH should be recovered from BGPsec_PATH correctly. [pCount]
- AS Confederation
  - AS numbers are private numbers which shouldn't appear in the eBGP AS_PATH.
  - What if an BGP router that don't support Transitive BGPsec or BGPsec?
    - RD and RF support Transitive BGPsec, but RE doesn't support Transitive BGPsec or BGPsec.
    - RD may generate a BGPsec_PATH that contains private AS Number.
- Can we remove BGPsec support in AS confederations? I.e., don't generate SIGNATURE for private AS numbers.



Q4: Route Aggregates

- Why it should avoid route aggregates?
  - BGPsec only support encapsulates one route.
  - Routes are signed into signature as well. SIG = F{P, ...}
  - If route aggregates occur, legacy BGP speakers may incorrectly process the BGP update, even including the following Transitive BGPsec speakers.

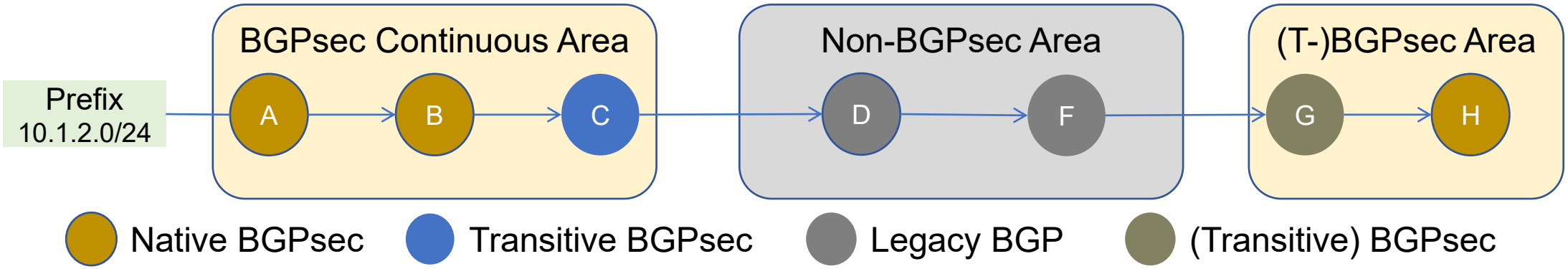


- [RFC 9774 - Deprecation of AS_SET and AS_CONFED_SET in BGP]

  can partly solve this problem.

- Can we set a new Community attribute to distinguish different routes?
  - Community is also an optional transitive attribute.
  - Each prefix consumes a Community value in an AS or we just define a new well-known Community value to indicate NO_ROUTE_AGGREGATES.

Q5: Compatibility with Native BGPsec



- BGP route propagation
  - AS A=>B: BGPsec UPDATE<P; BGPsec_Path{AS_A, SIG_{AB}}>
  - AS B=>C: BGPsec UPDATE<P; BGPsec_Path{AS_B, AS_A, SIG_{BC}, SIG_{AB}}>
  - AS C=>D: BGP UPDATE<P; AS_PATH{AS_C, AS_B, AS_A}, BGPsec_PATH{AS_B, AS_A, SIG_{BC}, SIG_{AB}}>
  - AS D=>F: BGP UPDATE<P; AS_PATH{AS_D, AS_C, AS_B, AS_A}, BGPsec_PATH{AS_B, AS_A, SIG_{BC}, SIG_{AB}}>
  - AS F=>G: BGP UPDATE<P; AS_PATH{AS_F, AS_D, AS_C, AS_B, AS_A}, BGPsec_PATH{AS_B, AS_A, SIG_{BC}, SIG_{AB}}>
- If AS G deploys Transitive BGPsec
  - AS G=>H: BGPsec UPDATE<P; BGPsec_PATH{AS_G, AS_F, AS_D, AS_C, AS_B, AS_A, SIG_{GH}, SIG_{BC}, SIG_{AB}}
    - But, downgrading to BGP may be a better choice for AS G, as it knows AS H only supports BGPsec.
- If AS G deploys Native BGPsec
  - The verification result of BGPsec_PATH is INVALID at AS G.

Conclusion

- It is difficult/infeasible to render BGPsec as a transitive approach for traversing undeployment areas while:
  - still using BGPsec_PATH type code in BGP UPDATE message,
  - and maintaining compatibility with native BGPsec.

We probably need a new optional, transitive path attribute so that it will not confuse with the BGPsec_PATH.

A bigger question: what should we, as a community, do to secure BGP paths?

Questions? Feedback?

Comments are welcomed.

Please email feedback to

draft-wang-sidrops-fcbgp-protocol@ietf.org

Thank you!

FC-BGP

- FC-BGP is Forwarding Commitment BGP

https://datatracker.ietf.org/doc/draft-wang-sidrops-fcbgp-protocol/

- It is an Optional, Transitive BGP Path attribute
  - It can keep compatible with BGPsec as they don't use the same Type Code.
- It still keeps AS_PATH in BGP UPDATE
  - So, it can keep compatible with BGP naturally.
- It uses a triple <PreviousAS, CurrentAS, NexthopAS> instead of the whole AS path in the signature.
  - No copy & paste attack can success. We have proved that.