Connection ID Design Team

A.K.A. WHAT'S THIS THING CALLED AGAIN?

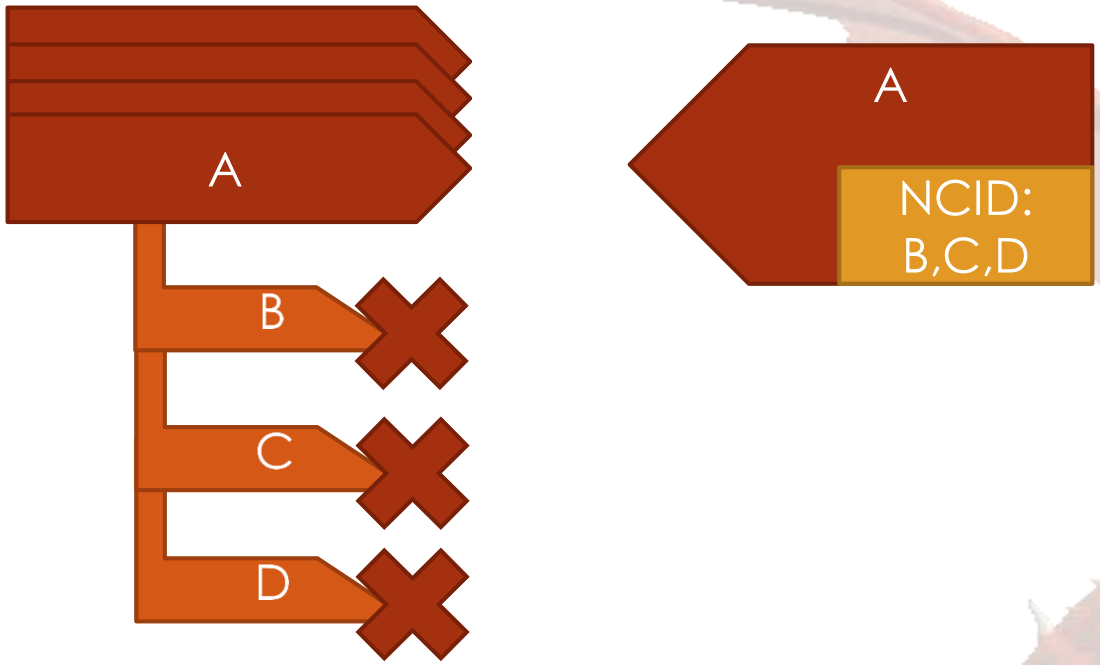
Sequence in -13+

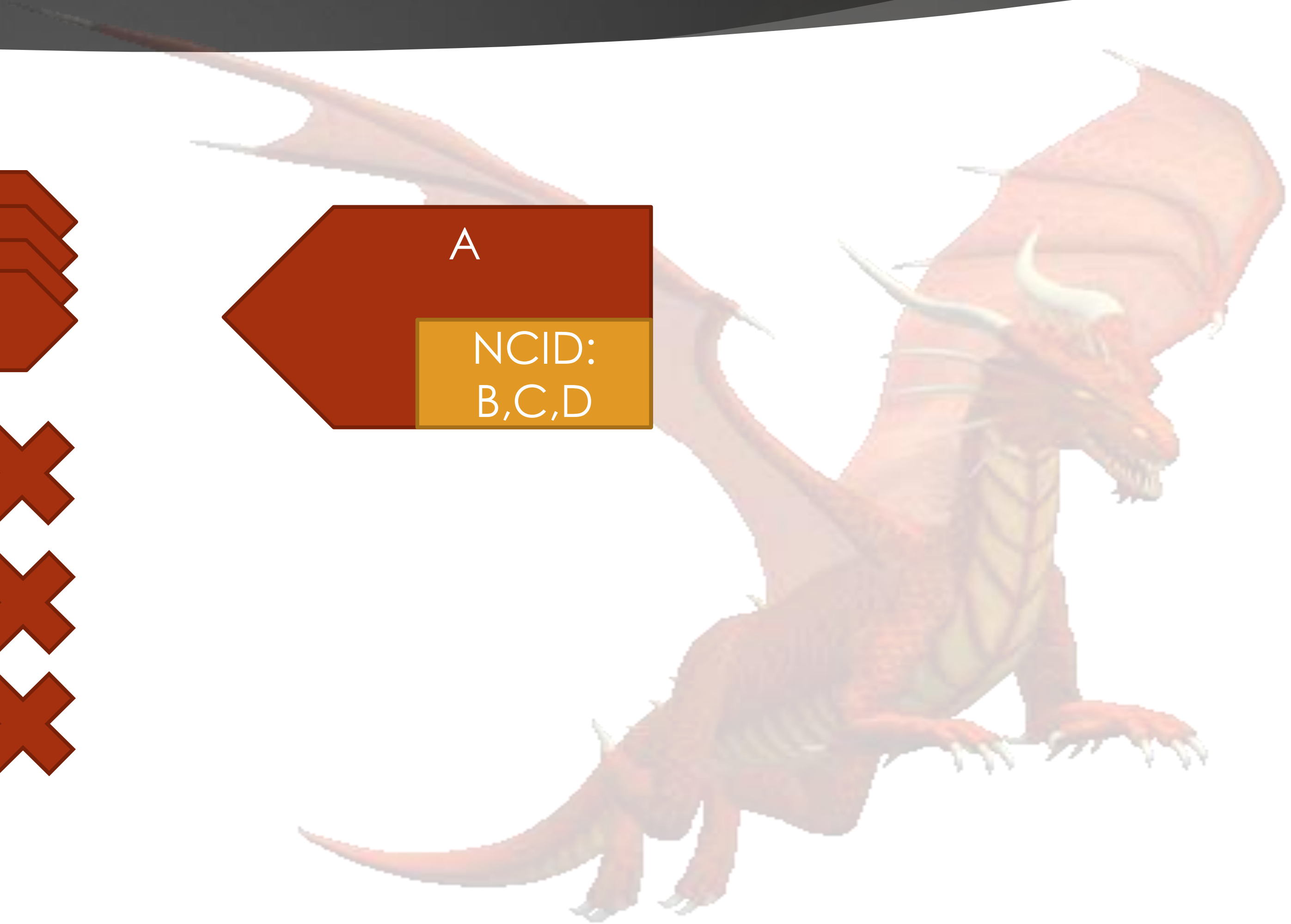
Seq.	CID	Token
-1	(A)	F(A)
0	(B)	F(B)
1	(C)	F(C)
2	(D)	F(D)
3	(E)	F(E)

Sequence without Gaps (-13)

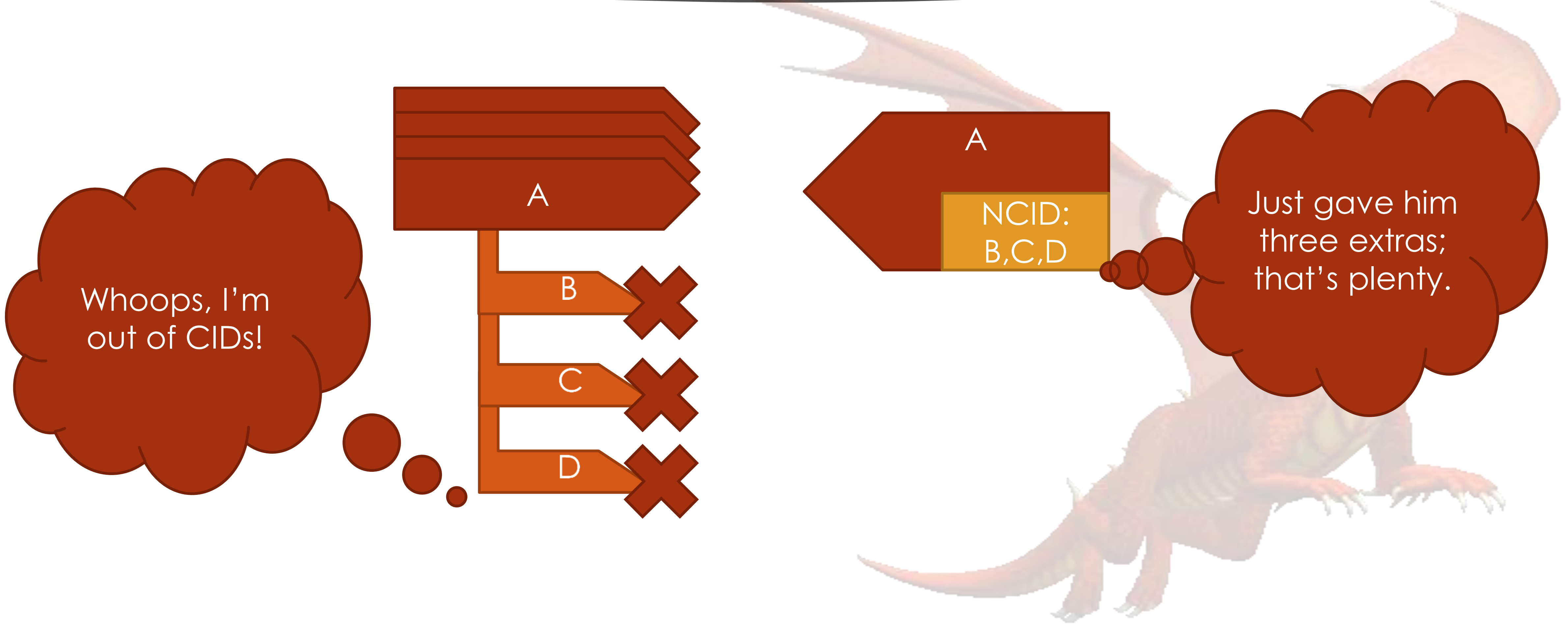
- No HoLB, because no packet number gaps
- Easier to specify behavior:
  - Use a higher sequence number than ever before when starting a new path
  - On each path, never use a sequence number less than the highest you've ever sent or received on that path

Here be dragons….





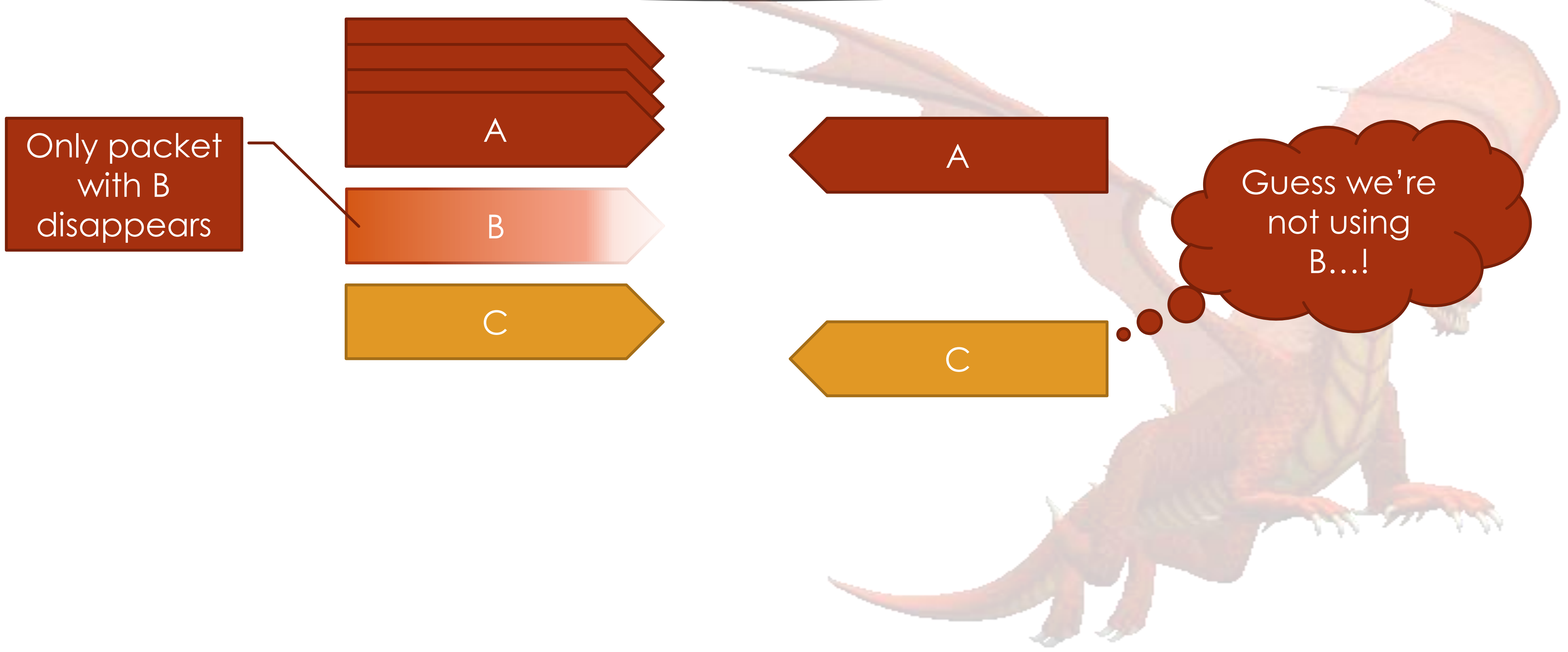
Here be dragons….



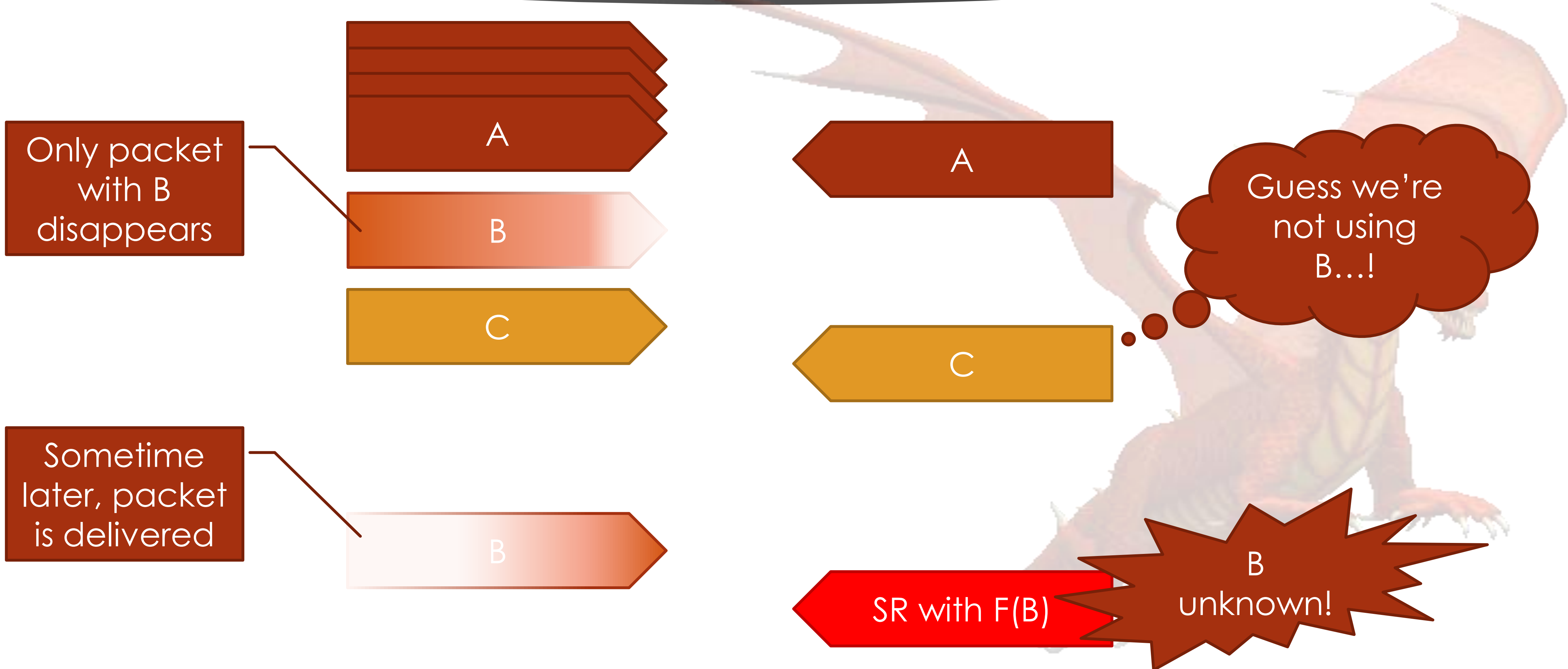
Raises some questions….

- It's possible to become unclear whether a peer has actually used a CID you've issued
- Given that, **how do I know when the peer needs more CIDs?**

Here be dragons….

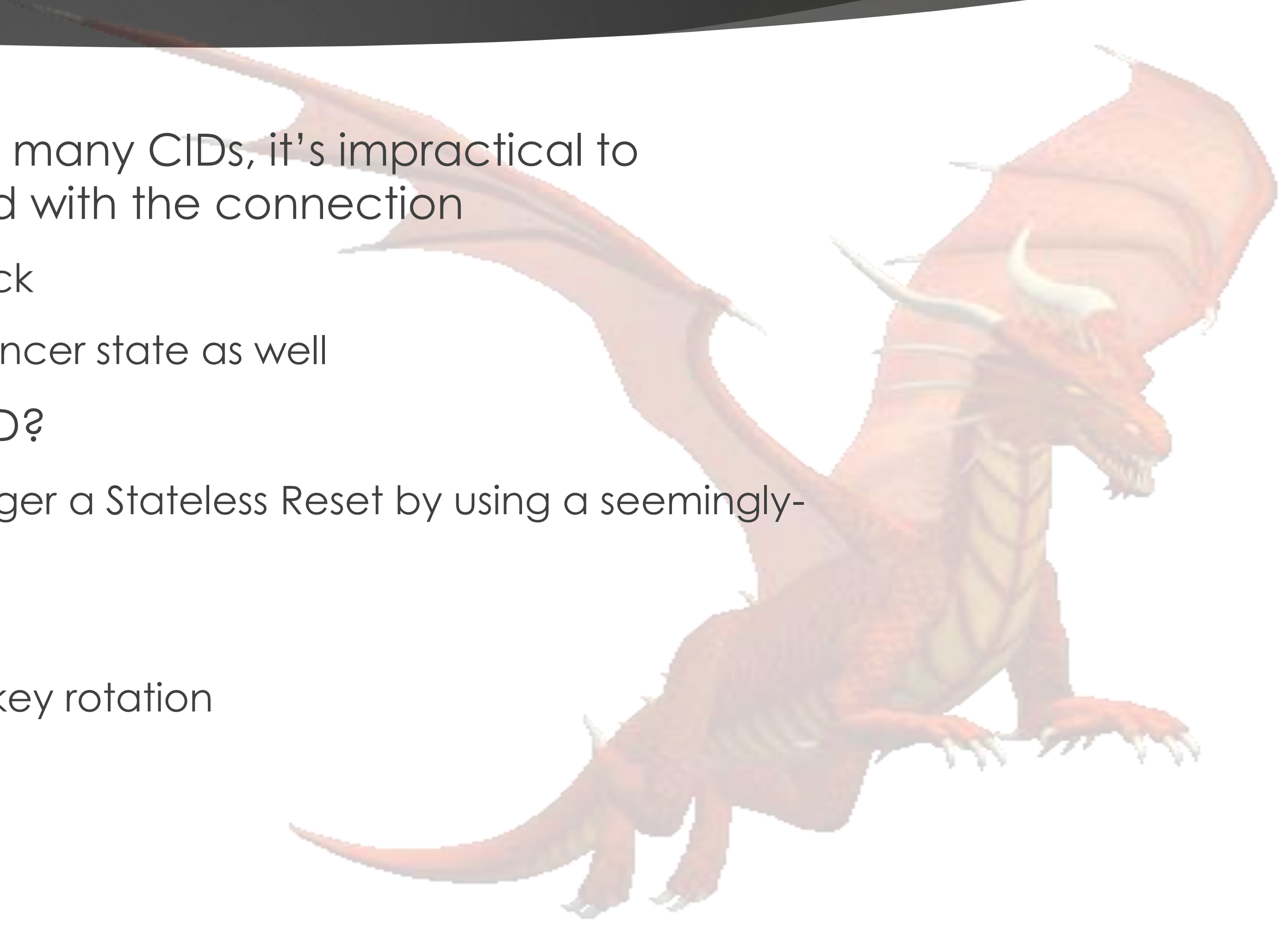


Here be dragons….


Only packet with B disappears
A
B
C
A
C
Guess we're not using B…!
Sometime later, packet is delivered
B
SR with F(B)
B unknown!

Raises some questions….

- Over a long-lived connection with many CIDs, it's impractical to remember all CIDs ever associated with the connection
  - Potential memory exhaustion attack
  - Might require allocating load balancer state as well
- But when is it safe to "forget" a CID?
  - Forget too early and peer can trigger a Stateless Reset by using a seemingly-valid CID
- Circumstances where CIDs expire
  - CID with encrypted payload and key rotation



Proposal: Frames

- NEW_CONNECTION_ID frame
  - Declares a new CID which can be used for the connection
  - **CIDs are non-revocable – once issued, valid until peer releases**
- RETIRE_CONNECTION_ID frame
  - Sent by recipient of NCID to indicate that a CID will no longer be used
    - …and the Stateless Reset Token will no longer be acknowledged
  - Can be sent on a different path than the one where the CID was previously used

Proposal: Rolling Forward

	Same CID	Different CID
Same IP : Port	Trivially Linkable	Highly Linkable
Different IP : Port	Highly Linkable	Breaks Linkability

- Change when peer changes CID
  - Risk of looping
- Change when peer changes IP or port
  - Doesn’t help when peer doesn’t change port

Proposal: Rolling Forward

	Same CID	Different CID
Same IP : Port	Trivially Linkable	Highly Linkable
Different IP : Port	Highly Linkable	Breaks Linkability

- Change when peer changes CID
  - Risk of looping
- Change when peer changes IP or port
  - Doesn't help when peer doesn't change port

+ recommendation to change port

Issues Explicitly Not Addressed

- Revocation of CIDs
  - Primary case to need that is a connection spanning multiple rolls of the key used to generate CIDs
    - These keys should be very long-lived; closing connections or maintaining state for very old connections seems acceptable
  - CIDs from one path might not be useful on a different path
    - Might need to improve for future multipath
- Negotiation of CID Pool Size
  - If CID pool runs out, connection might close
  - Issuer of CIDs is potentially consuming state to maintain many CIDs at once
    - Must be able to limit number outstanding

Issues Explicitly Not Addressed

- Quick Issue / Expiry of CIDs and Retransmission
  - Duplicate packets confuse things:
    - Endpoint issues NEW_CID(A)
    - Peer sends RETIRE_CID(A)
    - Duplicate packet arrives with NEW_CID(A) again
  - Current text: Remember the retired CIDs for 3xRTO, hope duplicates don’t stretch longer than that
  - Another solution: Sequence numbers

TBD

- Retiring CID while Stateless Reset in flight
  - If you forget the Token as soon as RETIRE_CID is sent, a Stateless Reset currently in flight won't have any effect
    - Probably okay – will trigger another Stateless Reset soon enough
  - Remembering the Token for a while (3xRTO?) consumes state, but makes the SR surface faster

Design Team Members

- Mike Bishop
- Eric Kinnear
- Kazuho Oku
- EKR
- Martin Thomson