IP Security Maintenance and Extensions (IPsecME) WG

IETF 112, Monday, November 8th, 2021

Chairs: Tero Kivinen
Yoav Nir

Responsible AD: Benjamin Kaduk

Note Well

This is a reminder of IETF policies in effect on various topics such as patents or code of conduct. It is only meant to point you in the right direction. Exceptions may apply. The IETF's patent policy and the definition of an IETF "contribution" and "participation" are set forth in BCP 79; please read it carefully.

As a reminder:

• By participating in the IETF, you agree to follow IETF processes and policies.

• If you are aware that any IETF contribution is covered by patents or patent applications that are owned or controlled by you or your sponsor, you must disclose that fact, or not participate in the discussion.

• As a participant in or attendee to any IETF activity you acknowledge that written, audio, video, and photographic records of meetings may be made public.

• Personal information that you provide to IETF will be handled in accordance with the IETF Privacy Statement.

• As a participant or attendee, you agree to work respectfully with other participants; please contact the ombudsteam (https://www.ietf.org/contact/ombudsteam/) if you have questions or concerns about this.

Definitive information is in the documents listed below and other IETF BCPs. For advice, please talk to WG chairs or ADs:
•BCP 9 (Internet Standards Process)
•BCP 25 (Working Group processes)
•BCP 25 (Anti-Harassment Procedures)
•BCP 54 (Code of Conduct)
•BCP 78 (Copyright)
•BCP 79 (Patents, Participation)
•https://www.ietf.org/privacy-policy/ (Privacy Policy)

Administrative Tasks

Bluesheets

We need volunteers to be:

- Two note takers
- One jabber scribe

Jabber: xmpp:ipsecme@jabber.ietf.org?join

MeetEcho: https://meetings.conf.meetecho.com/ietf112/?group=ipsecme&short=&item=1

Notes: https://codimd.ietf.org/notes-ietf-112-ipsecme

Agenda

- Note Well, technical difficulties and agenda bashing – Chairs (5 min) (12:00-12:05)
- Document Status – Chairs (10 min) (12:05-12:15)
- Work items
  - IPTFS – Christian Hopps (20 min) (12:15-12:35)
  - Quantum-resistent IKEv2 and big keys – Stefan-Lukas Gazdag (10 min) (12:35-12:45)
  - Group Key Management using IKEv2 – Valery Smyslov (10 min) (12:45-12:55)
  - Announcing Supported Authentication Methods in IKEv2 – Valery Smyslov (10 min) (12:55-13:05)
- AOB + Open Mic (55 min) (13:05-14:00)

WG Status Report

Publication requested:

- draft-ietf-ipsecme-ikev2-intermediate

Waiting for write-up / Chair review:

- draft-hopps-ipsecme-iptfs
- draft-fedyk-ipsecme-yang-iptfs
- draft-ietf-ipsecme-mib-iptfs
- draft-ietf-ipsecme-ikev2-multiple-ke
- draft-ietf-ipsecme-ikev1-algo-to-historic
- draft-ietf-ipsecme-labeled-ipsec

Work in progress:

- draft-ietf-ipsecme-g-ikev2
- draft-ietf-ipsecme-rfc8229bis

More detailed status of drafts in progress

- Group Key Management using IKEv2
  - draft-ietf-ipsecme-g-ikev2
  - Need more reviews
- Announcing Supported Authentication Methods in IKEv2
  - draft-smyslov-ipsecme-ikev2-auth-announce
  - Should be ready for WG adoption call
- TCP Encapsulation of IKE and IPsec Packets
  - draft-ietf-ipsecme-rfc8229bis
  - Ready for WGLC?

Presentations

- **IPTFS –**
  Christian Hopps
- Quantum-resistent IKEv2 and big keys –
  Stefan-Lukas Gazdag
- Group Key Management using IKEv2 –
  Valery Smyslov
- Announcing Supported Authentication Methods in IKEv2 –
  Valery Smyslov

Christian Hopps

LabN Consulting, LLC

IP Traffic Flow Security

Improving IPsec Traffic Flow Confidentiality

IETF 112 – “draft-ietf-ipsecme-iptfs-11”

2021 Recap

- WGLC Competed Feb 2021
  - Doc updated with received comments Feb 22 (-07)
- Post WGLC comments
  - Mar 30: Doc updated (-08) revised language (IPTFS->AGGFRAG) from Valery
  - Apr 5: Draft Write-Up submitted to Shepherd/Chairs
  - July 5: Doc updated (-09) clarifying that reorder window should be small, and should NOT force the replay window to be small as well
  - Sep 3: Doc updated (-10) recommending use of drop timer instead of reorder window to avoid long delays
    - Intending to address important comment from Tero
  - Oct 24: Doc updated (-11) took a guess at text Tero would accept WRT optionally sending immediately out-of-order
  - Oct 31: Text from Tero – one last outstanding issue based on this text

Last Issue To Resolve

- Update -11 added text saying the receiver **MAY** optionally send whole inner packets on receipt w/o waiting for earlier misordered tunnel packets to arrive.
- Tero's alternate text has same mechanism ***but*** changes it to "**SHOULD**", restoring the original in-order delivery as a **MAY**.

New text in -11

“As an optional optimization (e.g., to handle very lossy and/or reordered tunnel paths), the receiver **MAY** transmit any fully formed inner packets contained within the AGGFRAG_PAYLOADs prior to re-ordering the outer packets.”

Proposed Tero Text

The receiver **SHOULD** process incoming AGGFRAG_PAYLOAD payloads as soon as they arrive as much as it can. I.e., if the incoming AGGFRAG_PAYLOAD packet contains complete inner packet(s), receiver should extract them and forward them immediately. For partial packets the receiver needs to keep the partial packets in the memory until the they fall out from the reordering window, or until the missing parts of the packets is received, in which case it will reassemble them and send them out. If AGGFRAG_PAYLOAD payload contains multiple packets they SHOULD be sent out in the order they are in the AGGFRAG_PAYLOAD (i.e., keep the original order they were received on the other end).

… [reworded original text]

Counter and Compromise Proposal

- Lou Berger suggested on list, swapping SHOULD/MAYs
  - In-order delivery (which might incur a small delay) remains recommended

    "FWIW I'm basing my comments on my routing area experience where a huge amount of work has been put into maintaining ordering experienced by user traffic at significant implementation expense, i.e., in support of ECMP and other multipath solutions in protocols and hardware."
  - Out-of-order delivery still allowed
  - I.e., adopt Tero's text ***but*** keep original as the recommended behavior
- Lou's mail also OK with both **MAYs** with a configuration selection

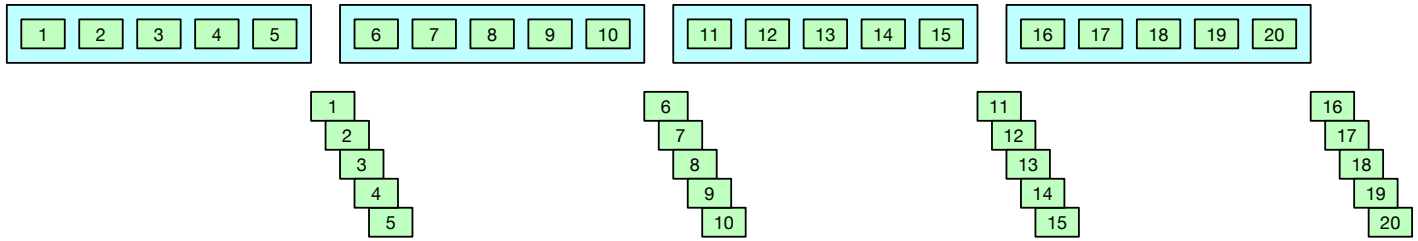
Issue with Send Immediately

- Amplifies end-user experienced misordering
- Routers are built to not introduce misordering or bizarre delays in packet flows

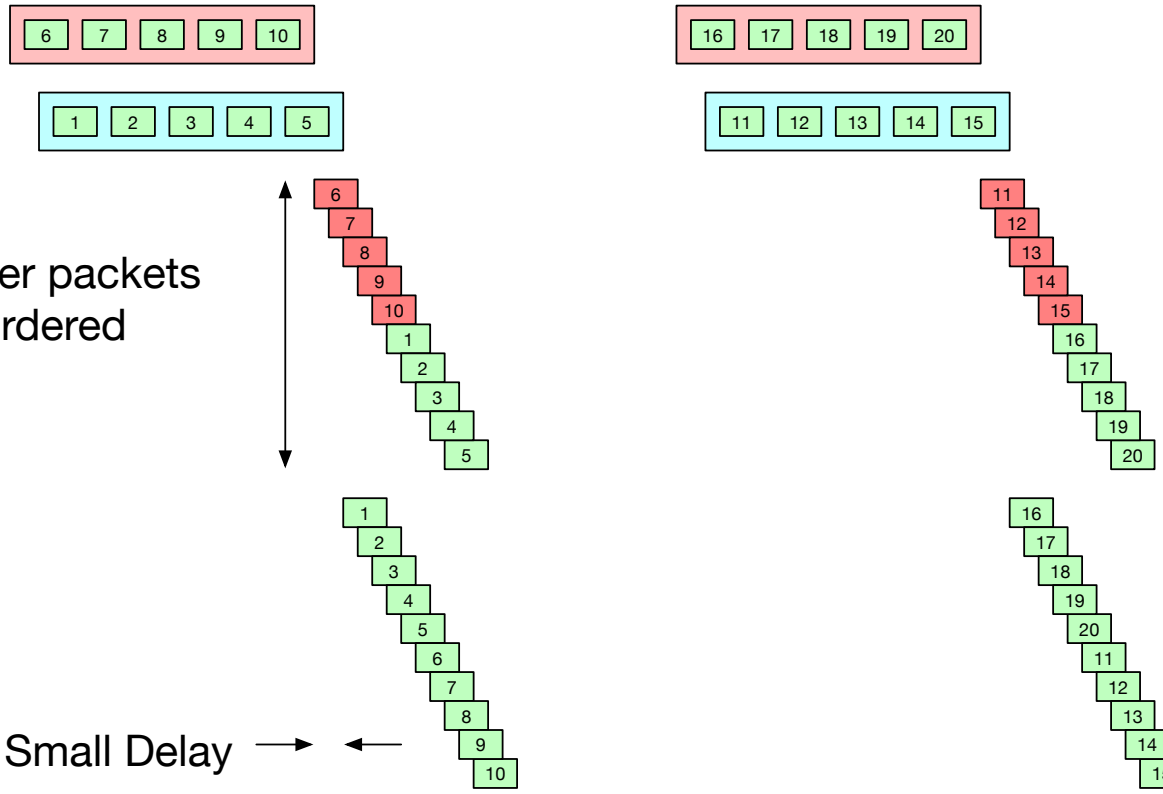
Why reordering outer packets “Just Works”

- Operationally significant delays unlikely from misordering
  - At high send rate (e.g., line rate, no send gap)
    - A reasonable reordering window won’t introduce unreasonable delay to correct ordering
  - At lower send rate (wide sending gap)
    - Misordered sloths are simply dropped
    - Drop timer limits any delay due to these drops

Ordered Outer



Misordered Outer



For Discussion: MAY vs SHOULD

The receiver **MAY|SHOULD** process incoming AGGFRAG_PAYLOAD payloads as soon as they arrive as much as it can. I.e., if the incoming AGGFRAG_PAYLOAD packet contains complete inner packet(s), receiver should extract them and forward them immediately. For partial packets the receiver needs to keep the partial packets in the memory until the they fall out from the reordering window, or until the missing parts of the packets is received, in which case it will reassemble them and send them out. If AGGFRAG_PAYLOAD payload contains multiple packets they SHOULD be sent out in the order they are in the AGGFRAG_PAYLOAD (i.e., keep the original order they were received on the other end).

Instead of the method described in the previous paragraph the receiver **SHOULD|MAY** reorder out-of-order AGGFRAG_PAYLOAD payloads received into in-sequence-order AGGFRAG_PAYLOAD payloads (Section 2.2.3),and only after it has in-order AGGFRAG_PAYLOAD payload stream,receiver extracts the inner-packets. In this case the receiver considers a packet lost when ***the drop timer expires*** or it's sequence number is abandoned (e.g., pushed out of the re-ordering window~~, or timed-out~~) by the reordering algorithm. Using this method will make sure the packets are sent in-order, i.e., there is no reordering possible, but the cost is that any lost packet will cause delay of ***the drop timer interval*** ~~full reorder window~~, and there will be extra burstiness in the output stream (when lost packet is dropped out from the re-order window, all outer packets received after that are then immediately processed, and sent out back to back).

Next Steps

- Publish document based on today's discussion/resolution
- No other issues
- Submit to IESG for publication

IPTFS Reorder/lost frame issue

Tero Kivinen <kivinen@iki.fi>

Section 2.5 of IPTFS draft:

2.5. Summary of Receiver Processing

An AGGFRAG enabled SA receiver has a few tasks to perform.

The receiver first reorders, possibly out-of-order ESP packets received on an SA into in-sequence-order AGGFRAG_PAYLOAD payloads (Section 2.2.3). If congestion control is enabled, the receiver considers a packet lost when it's sequence number is abandoned (e.g., pushed out of the re-ordering window, or timed-out) by the reordering algorithm. As an optional optimization (e.g., to handle very lossy and/or reordered tunnel paths), the receiver MAY transmit any fully formed inner packets contained within the AGGFRAG_PAYLOADs prior to re-ordering the outer packets.

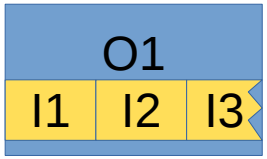
Additionally, if congestion control is enabled, the receiver sends congestion control data (Section 6.1.2) back to the sender as described in Section 2.4.2 and Section 3.

Finally, the receiver processes the now in-order AGGFRAG_PAYLOAD payload stream to extract the inner-packets (Section 2.2.3, Section 6.1).

Issues in section 2.5

- It implies that normal processing is to reorder outer ESP packets to in-order stream and process them after that.
- New text was added in -11 version to allow optimization where receiver MAY transmit any fully formed inner packets before re-ordering.
- I think this optimization should be default, and the in-order processing should not be used in normal cases.

Normal flow



Receiver receives O1, starts processing it.

Outer packet

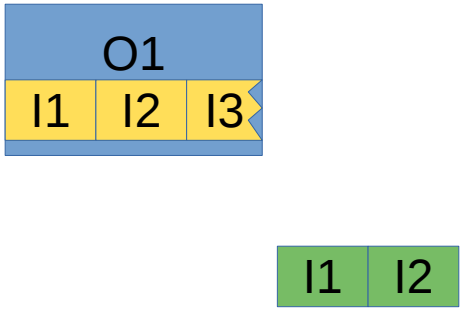
Complete inner packet

Start of the inner packet

End of the inner packet

Inner packet out from the device

Normal flow



Sends out inner packets I1 and I2, cannot send I3, as it is not fully received

Ox — Outer packet

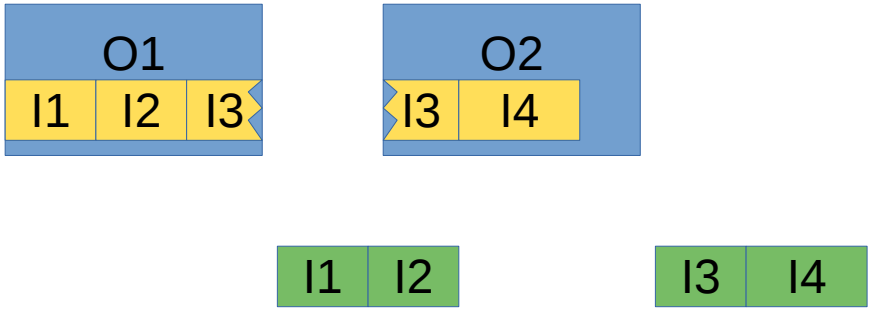
Ix — Complete inner packet

Ix — Start of the inner packet

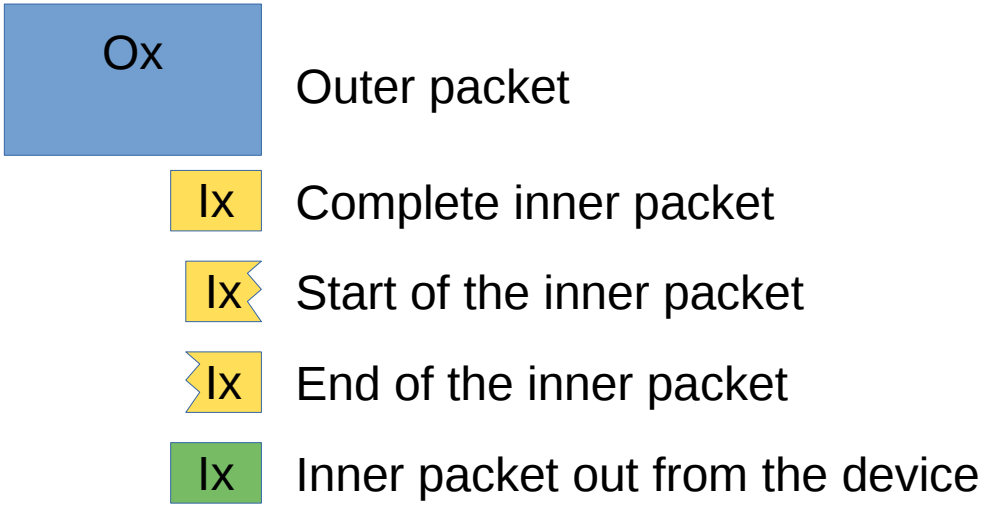
Ix — End of the inner packet

Ix — Inner packet out from the device

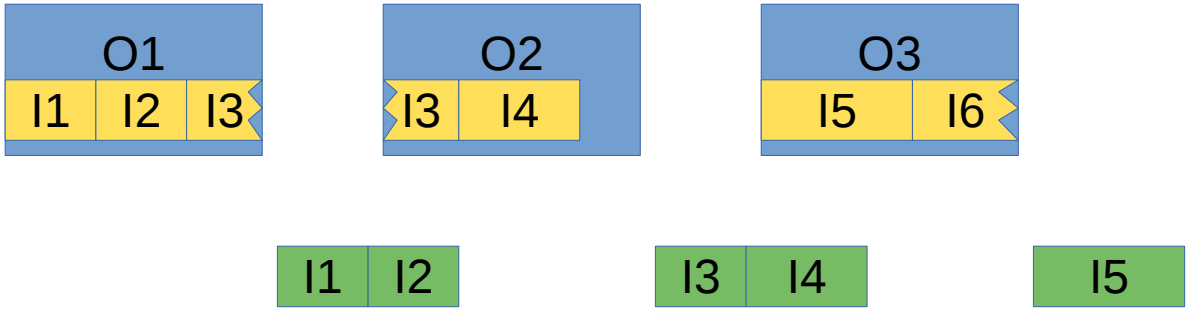
Normal flow



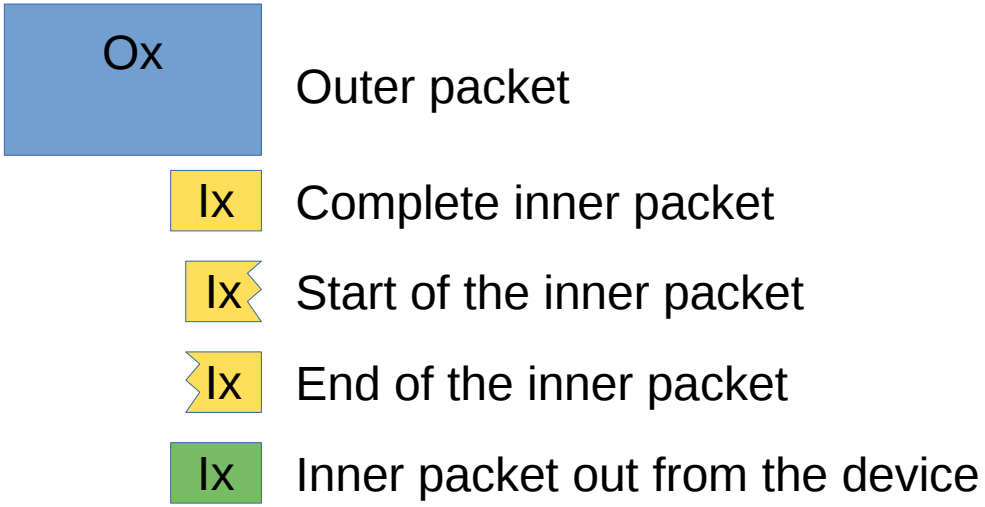
Receiver receives O2, sends out nowcomplete I3, and I4



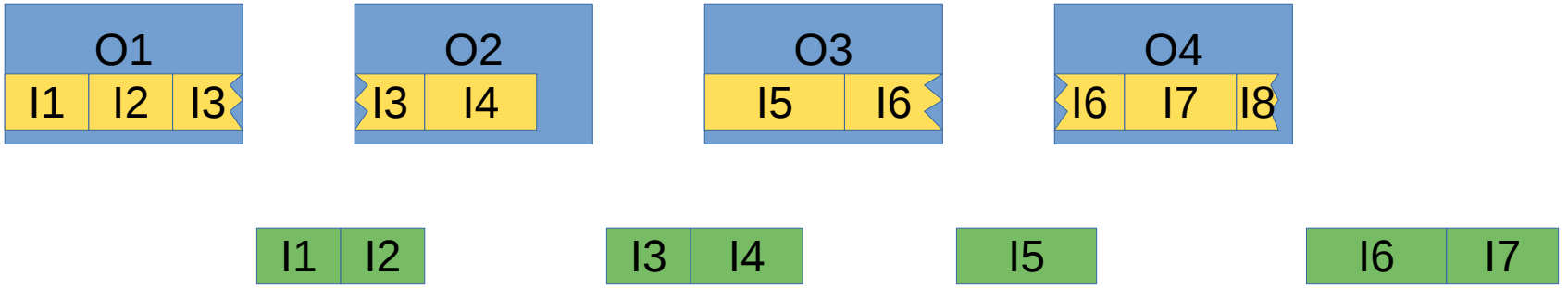
Normal flow



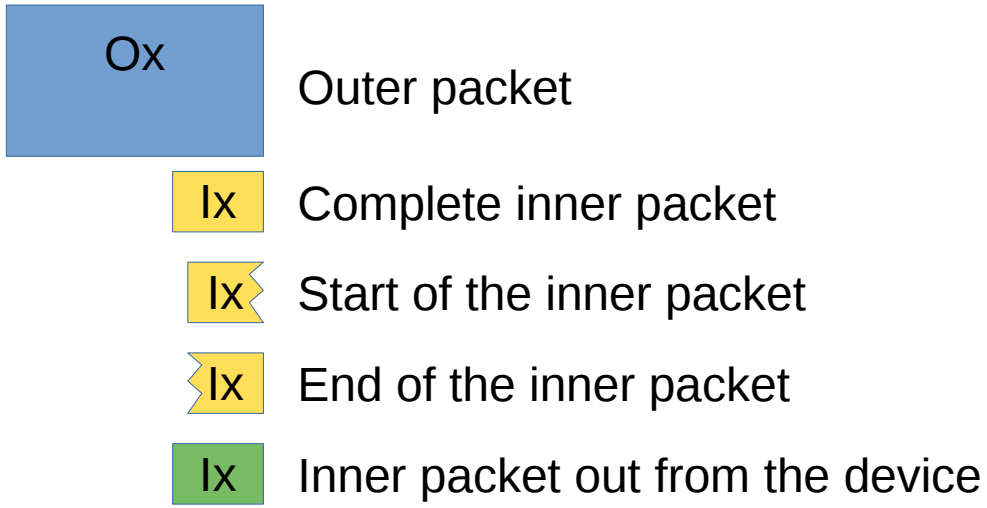
Receiver receives O3, Sends out I5



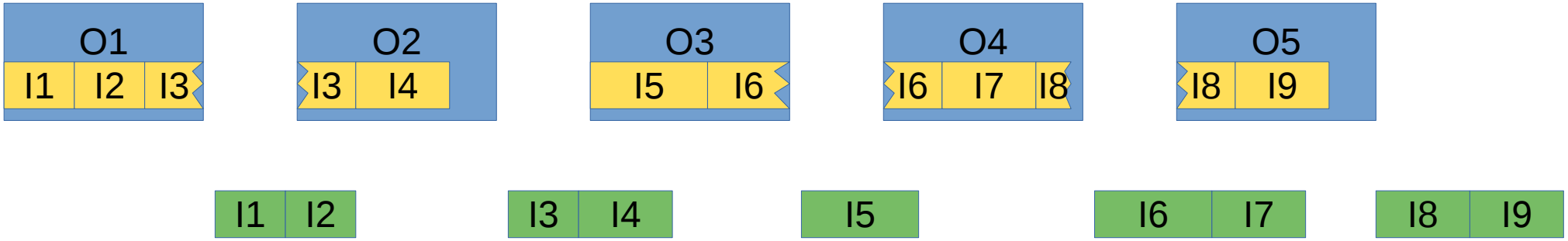
Normal flow



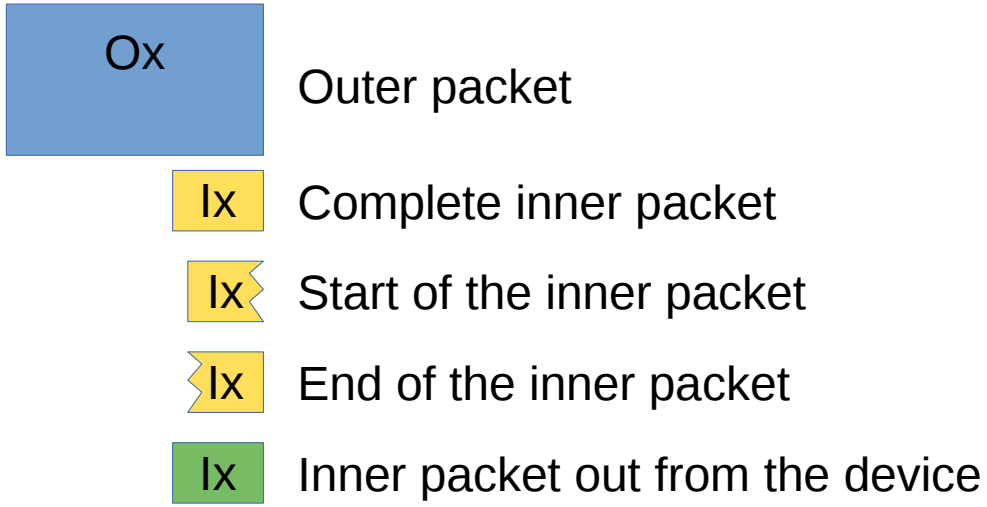
Receiver receives O4, Sends out I6, and I7



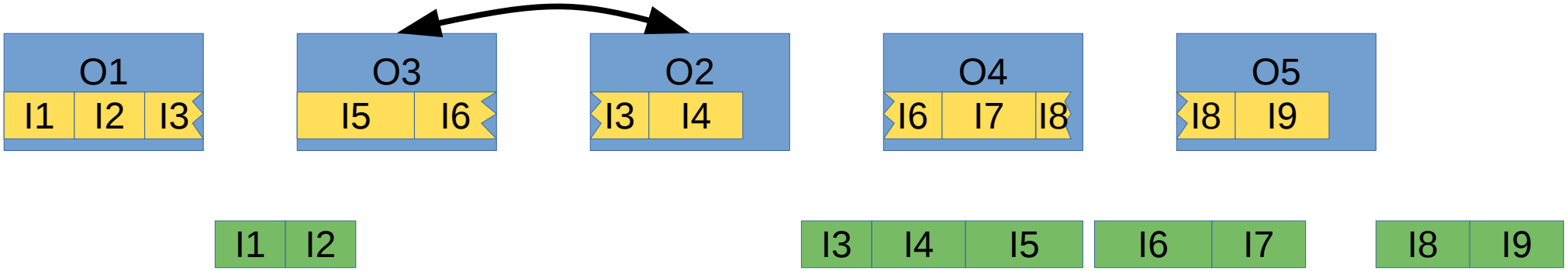
Normal flow



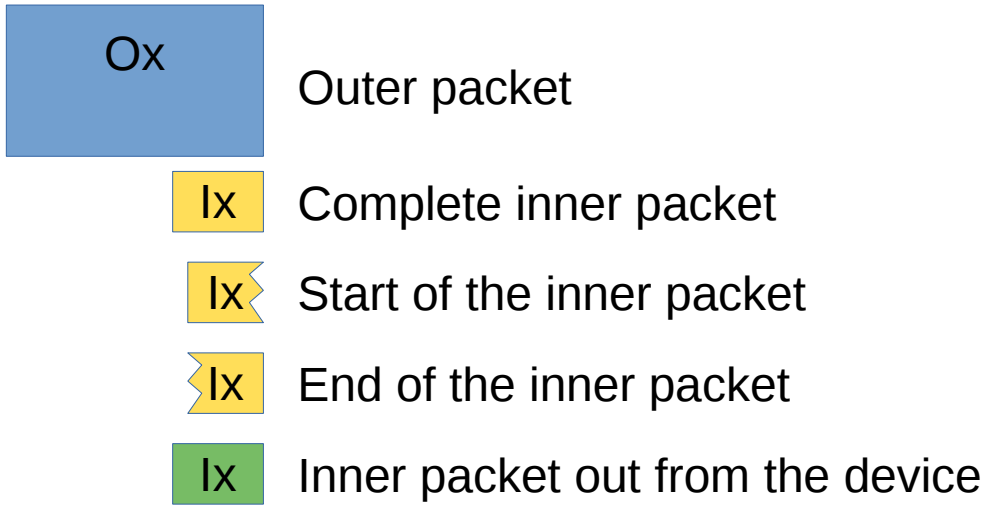
Receiver receives O5, Sends out I8, and I9



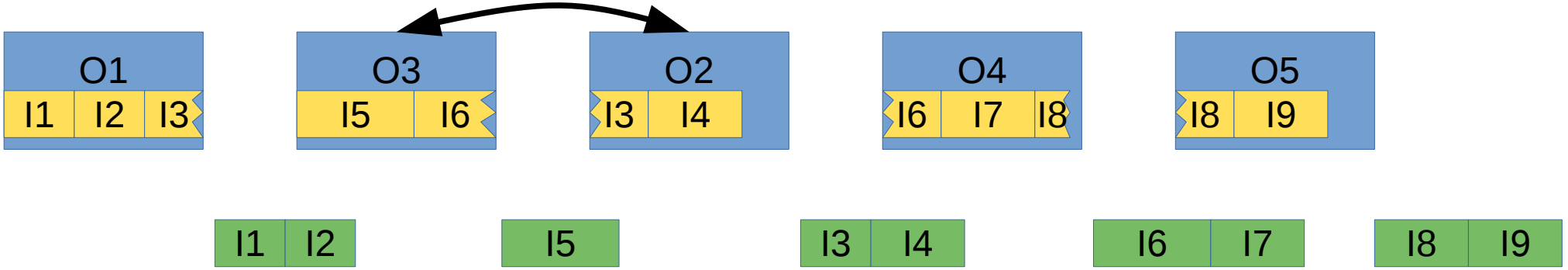
Reordered flow (in-order)



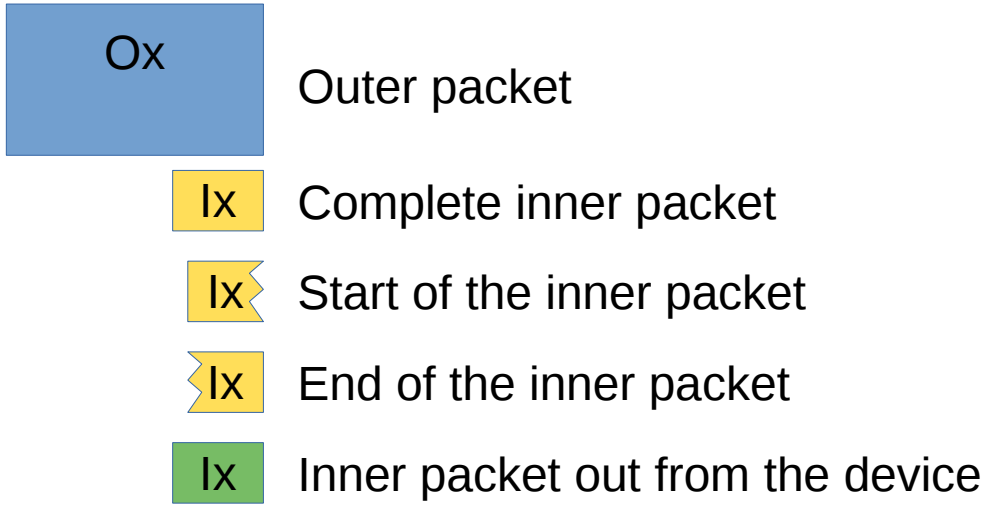
When receiver receives O3, it buffers it. When it receives O2, it processes O2, and then O3, thus fully complete I5 will be delayed until O2 arrives to keep order.



Reordered flow (immediate)

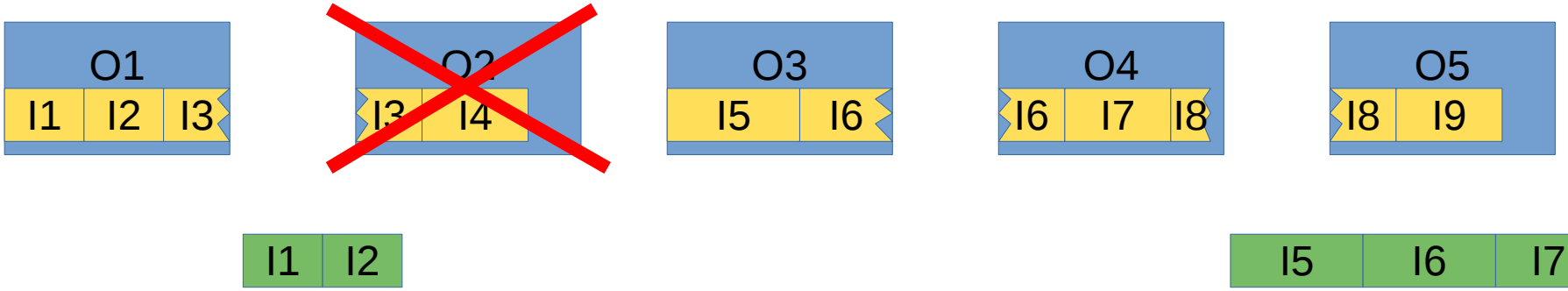


When receiver receives O3, it buffers I6, but sends I5 out as it is complete. When it receives O2, it reassembles I3, and sends out, and then sends I4. I5 is sent before I3 and I4, meaning the reordering in outer frames is kept and is visible in inner frames too.

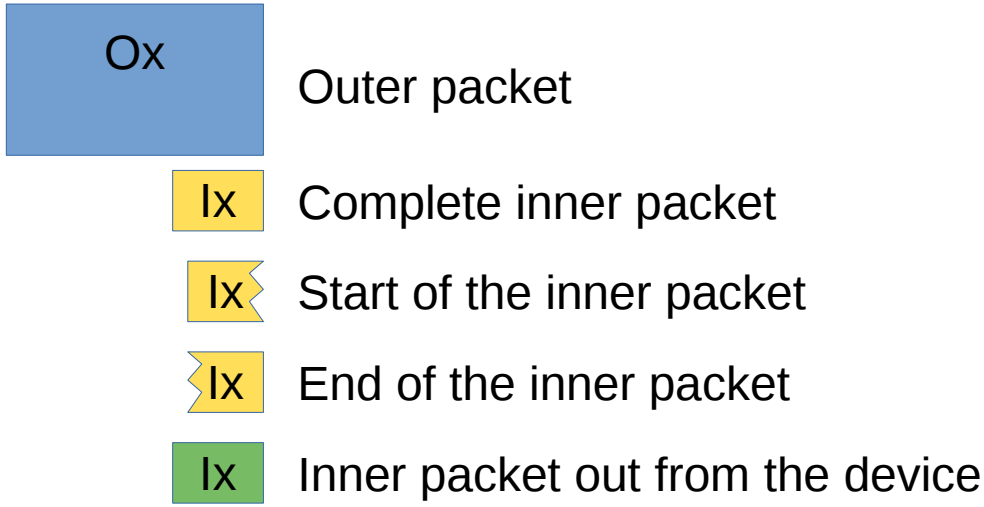


Lost frame (in-order)

Assuming reorder window of 2

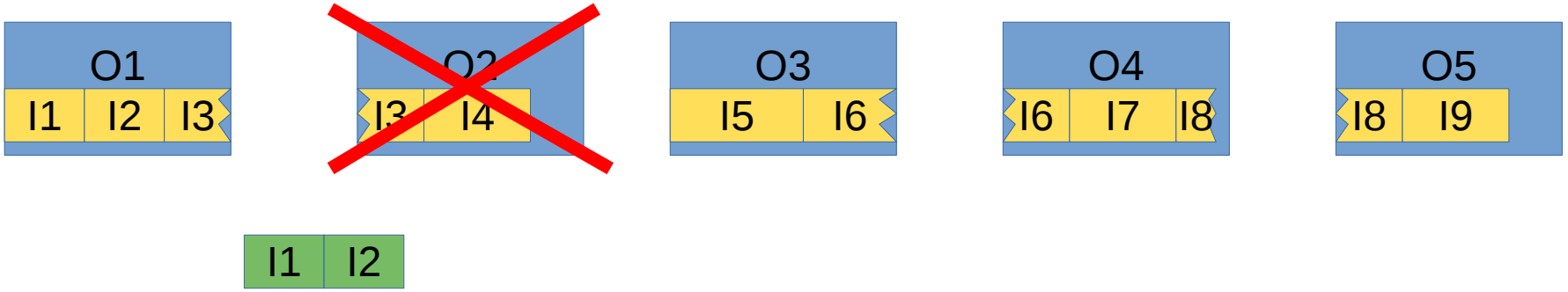


When receiver receives O3, it realises frame is missing, so it buffers O3 until reorder buffer gets full. If reorder window is 2, after receiving O4 it knows O2 was lost, and only after that it will send out I5, I6, and I7. Receiver needs to buffer I3 of O1, and O3 until they fall out from reorder window, thus reorder window greatly affects memory usage.

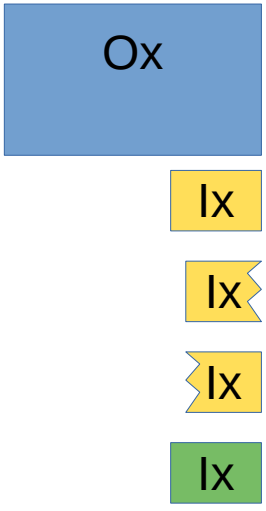


Lost frame (in-order)

Assuming reorder window of 3

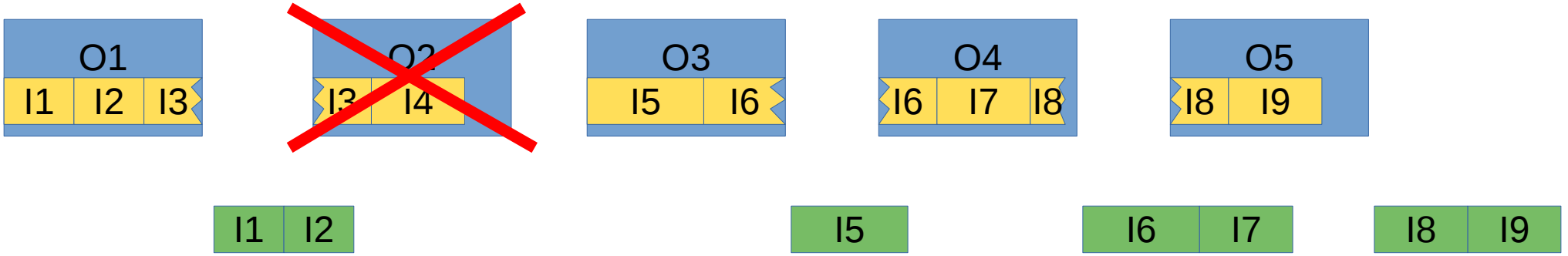


When receiver receives O3, it realises frame is missing, so it buffers O3 and O4 until reorder buffer gets full. If reorder window is 3, after receiving O5 it knows O2 was lost, and only after that it will send out I5, I6, I7, I8, and I9. Receiver had to buffer I3, O3, and O4. Larger the reorder window greater the delay and memory usage.

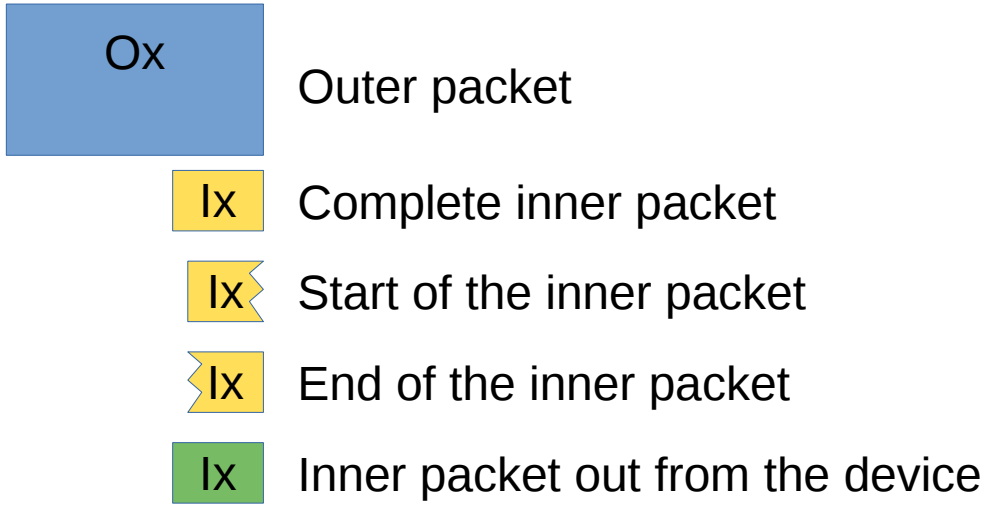


Lost frame (immediate)

Reorder window does not affect processing



When receiver receives O3, it processes it normally and sends out I5. It processes O4, and O5 normally and send the inner packets out when they are complete. No buffering happening. Large reorder window does not affect delay, and only I3 from O1 is buffered until O2 falls out from the reorder window and then it is discarded.

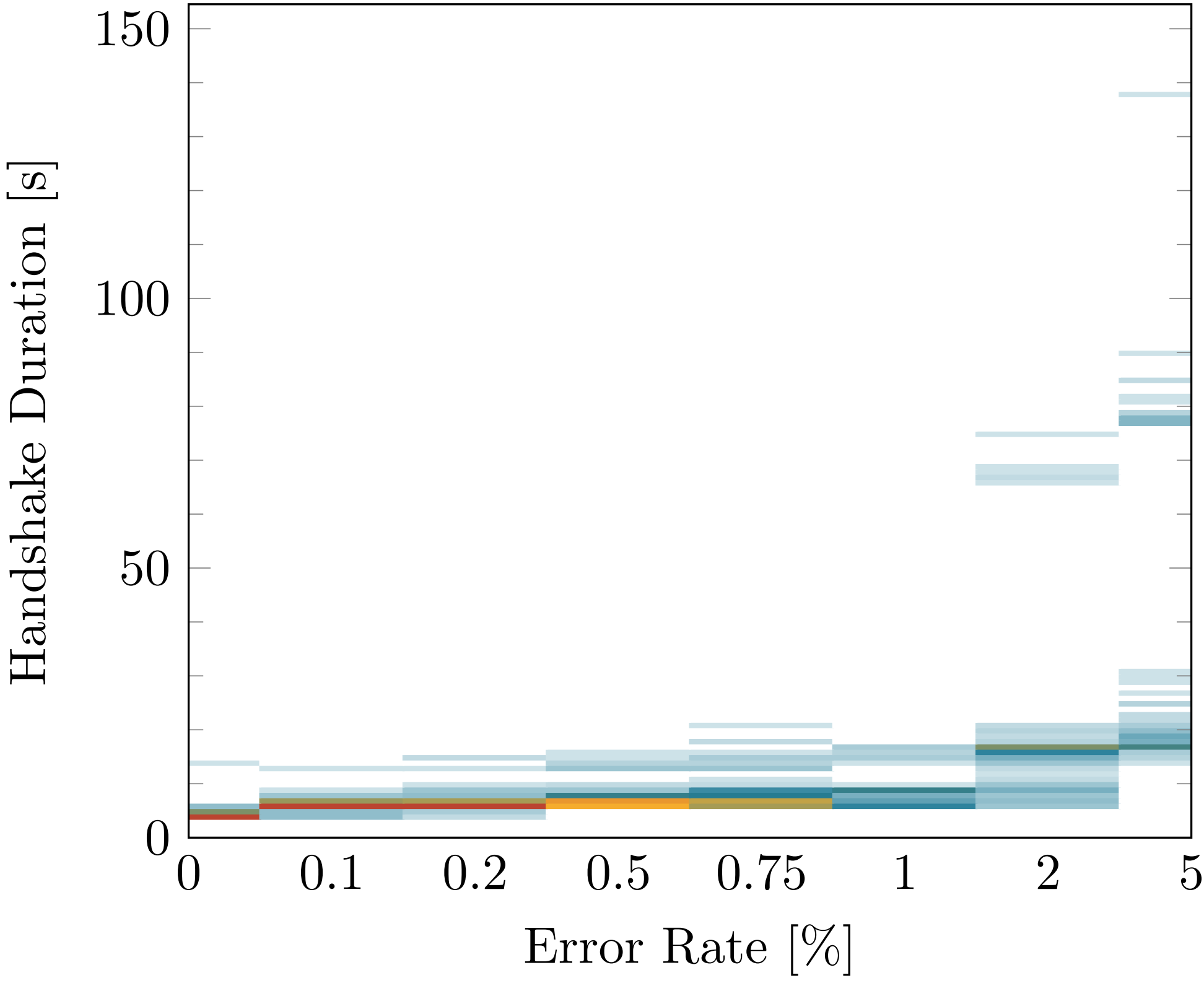


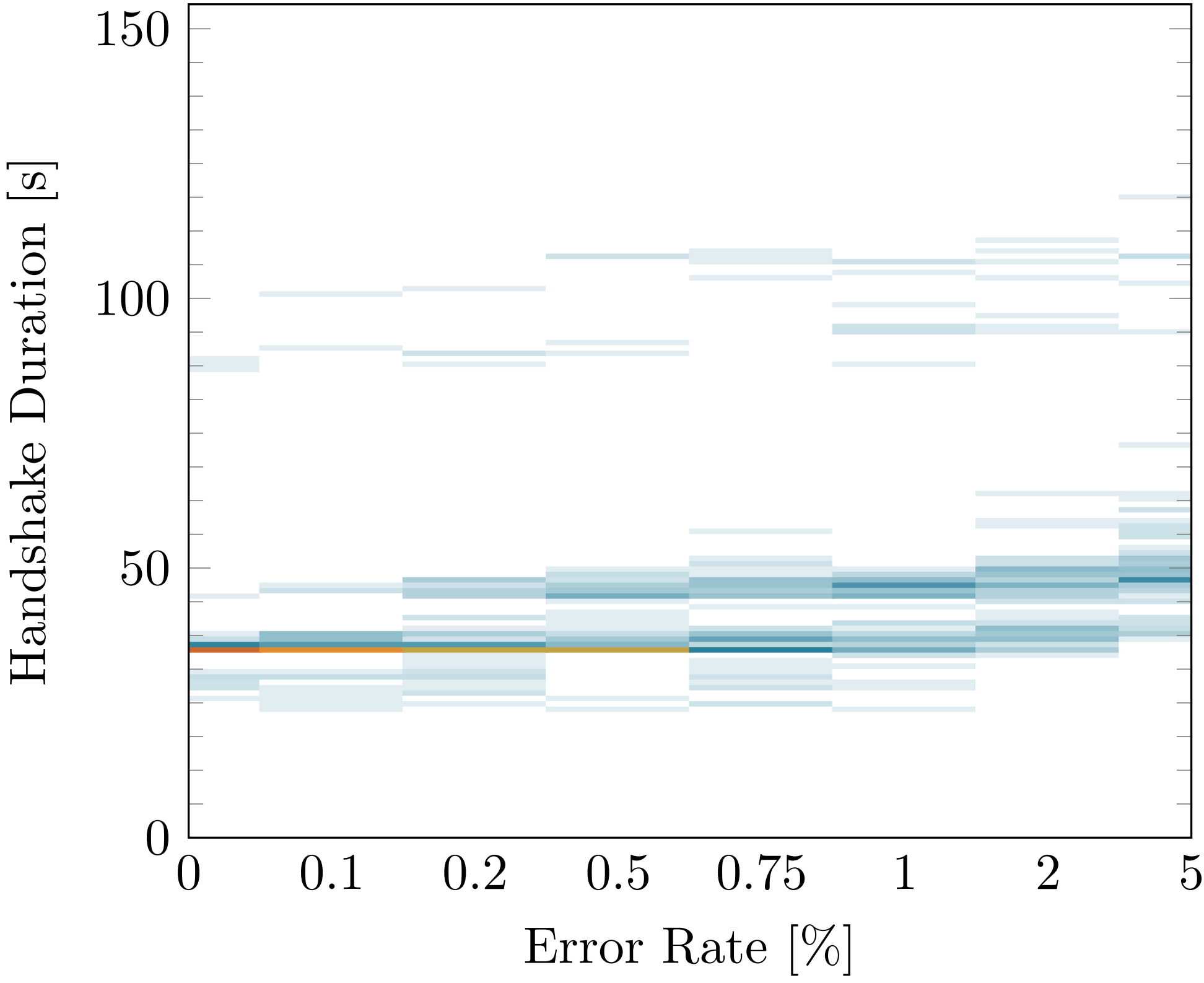
Presentations

- IPTFS –
  Christian Hopps
- **Quantum-resistent IKEv2 and big keys –
  Stefan-Lukas Gazdag**
- Group Key Management using IKEv2 –
  Valery Smyslov
- Announcing Supported Authentication Methods in IKEv2 –
  Valery Smyslov

Quantum-Resistant IKEv2 and Big Keys

Stefan-Lukas Gazdag, Daniel Herzinger


Handshake Duration [s]
150
100
50
0
0
0.1
0.2
0.5
0.75
1
2
5
Error Rate [%]


Handshake Duration [s]
150
100
50
0
0
0.1
0.2
0.5
0.75
1
2
5
Error Rate [%]

Presentations

- IPTFS –
  Christian Hopps
- Quantum-resistent IKEv2 and big keys –
  Stefan-Lukas Gazdag
- **Group Key Management using IKEv2 –
  Valery Smyslov**
- Announcing Supported Authentication Methods in IKEv2 –
  Valery Smyslov

Group Key Management using IKEv2

draft-ietf-ipsecme-g-ikev2

Valery Smyslov
ELVIS-PLUS

Brian Weis
Independent

IETF 112

Securing IP Multicast

- IP multicast applications
  - Contain at least 1 sender, and N receivers
  - Take advantage of the network to route and replicate IP packets, such that the same packet reaches all N receivers
- This requires senders and receivers to share setup an IPsec SA using the same keys
  - The IPsec policy and keys are not negotiated, but instead they are distributed by a Group Controller / Key Server (GCKS) to Group Members (GMs)
  - A GM invokes a unicast Registration protocol to authenticate to the GCKS. The GCKS then authorizes the GM, and distributes IPsec policy and keys to the GM.
  - A Rekey protocol enforces a time-based key rollover strategy

Distribution of Group Keys in IEEE 802.15

- IEEE 802.15.9 specified IKEv2 as one of KMPs for IEEE 802.15.4
  - IEEE Std 802.15.9-2015 left group keys distribution out of scope
- Draft 05 version of the IEEE Std 802.15.9 standard (March 2021) specifies that G-IKEv2 is used for group key distribution
  - GSA_INBAND_REKEY over unicast SA is used
  - SPI field in GSA payload is used to specify the type of group key

Document Status

- Has been in development for several years
  - few implementations of early draft versions exist
- Has been adopted by IPSECME WG in 2019
- Version -01 (July 2020): major rewrite
- Version -02 (January 2021): minor update
- Version -03 (July 2021): minor update
- For authors the draft looks mature
  - however, more reviews are needed

Outline of -01 Changes

- Policy representation changed
  - before: IKEv1 style, mostly using attributes
  - now: IKEv2 style – using transforms, attributes are still used to represent variables
- Format of GSA and KD payloads changed
- Group keys representation changed
  - before: group keys were transferred in clear inside KD payload
  - now: all keys are encrypted inside KD payload, using either SK_d derived key or other group key
- LKH (Logical Key Hierarchy) is integrated in core G-IKEv2
  - before: dedicated attributes were used to transfer LKH keys
  - now: LKH functionality is integrated into the core G-IKEv2 protocol, GM semantics doesn't depend on key management method

Outline of -01 Changes (cont.)

- IANA considerations are rewritten
  - now it's more an extension to IKEv2 than a separate protocol (IKEv2 IANA registries are used)
  - many parameters have been renamed to better reflect their purpose
- A lot of clarifications
  - AUTH payload calculation for GSA_REKEY messages is described in details
  - introduced means to indicate cross-dependency of supported algorithms in SAg payload
  - using PPK in G-IKEv2 is clarified
  - using ESN is clarified (in -02)
  - failover in situations when rekey message was missed clarified (using NEXT_SPI)
  - example of using LKH is rewritten

GSA Payload

Contains policy necessary to participating in the group:

- Protocol (GIKE_REKEY, AH, ESP)
- Traffic Selector
- Transforms for algorithms and methods used in the policy
- Attributes for variables that change over time (like initial Message-ID)
- GSA format is now common for KEK (GIKE_REKEY) and TEK (AH, ESP)
  - GAP (Group Policy) shares the same format and is distinguished by zero protocol

KD Payload

Contains keying material necessary for the policy in the GSA payload:

- One or more keys are conveyed in the KD payload
- Security parameters are also conveyed in the KD payload
- Each key is individually wrapped in a new structure Wrapped Key
- Each Wrapped Key structure is encrypted using either SK_d derived key or other group key
- LKH capability is now integrated into G-IKEv2 core and is achieved by including several keys into the KD payload logically linked by encrypting next key in the tree with previous one
- Wrapped Keys may contain either group keys (common for a whole group or for subset of its members) or member keys (allows for provision keys for a member during GSA registration, needed for LKH)

IDg Payload

Contains identity of the group a GM wants to join (no changes since -00):

- has the same format as IKEv2 ID payload
- only some ID types are expected to be used
  - `ID_KEY_ID` MUST be supported
  - `ID_IPV4_ADDR`, `ID_IPV6_ADDR`, `ID_FQDN`, `ID_RFC822_ADDR` SHOULD be supported

Reused IKEv2 payloads

Payloads that have the same types as in IKEv2, but different semantics:

- SAg (GM Supported Transforms)
  - declares which Transforms a GM is willing to accept
  - has the same format as IKEv2 SA payload, but slightly different semantics, which allow to indicate inter-dependency of supported algorithms
- D (Delete Payload)
  - used when the GCKS may want to signal to group members to delete policy (e.g., data flows finished, change of policy)
  - semantics is slightly different from IKEv2, allowing to delete all SAs

New Notifications

- INVALID_GROUP_ID (error notify)
  - GCKS informs GM that the requested Group ID in a registration protocol is invalid
- AUTHORIZATION_FAILED (error notify)
  - GCKS informs GM that it is not authorized to join the requested Group ID
- REGISTRATION_FAILED (error notify)
  - GCKS informs GM that for some reason the GM cannot join the group
  - GM sends to GCKS to unregister from the group
- SENDER (status notify)
  - GM informs the GCKS about its intention to be a sender in the group
  - requests a number of Sender-ID values, that are used as part of a counter-mode transform nonce (RFC 6054)
- REKEY_IS_NEEDED (status notify) – added in -01
  - GCKS informs GM that it must rekey IKE SA before receiving sensitive information (used in PPK scenarios)

Reused IKEv2 Notifications

- USE_TRANSPORT_MODE
  - semantics is changed, so that Protocol and SPI fields are used to indicate which SA to create in transport mode
  - multiple instances can be sent if multiple SAs are being created

Thank you!

- Comments?
- Questions?
- Please review the document
  - WGLC?

Presentations

- IPTFS –
  Christian Hopps
- Quantum-resistent IKEv2 and big keys –
  Stefan-Lukas Gazdag
- Group Key Management using IKEv2 –
  Valery Smyslov
- **Announcing Supported Authentication Methods in IKEv2 –**
  Valery Smyslov

Announcing Supported Authentication Methods in IKEv2

draft-smyslov-ipsecme-ikev2-auth-announce

Valery Smyslov
svan@elvis.ru

IETF 112

Authentication in IKEv2

- Unlike IKEv1, authentication method in IKEv2 is not negotiated, each peer is free to use whichever method it thinks is appropriate
- Generally works well if there is only one way of doing authentication or there is no ambiguity in choosing among several of them
- If peers can use several methods to authenticate each other, it is possible that initiator selects authentication method unsupported by the responder
  - less likely in the opposite direction, but still possible

The Problem

- The problem was first encountered when RSA-PSS signature format appeared in IKEv2
  - newer initiators tried to use PSS signatures while older responders didn't support it, sending back **`AUTHENTICATION_FAILED`**
  - if initiators knew responders' capabilities they would have chosen PKCS#1 and the SA succeeded

Source of the Problem

- Currently there is no way for the peers to explicitly indicate the supported authentication methods
  - it is possible to guess them via indirect means, e.g. `CERTREQ` content, but this is unreliable
- With new signature formats and authentication methods appearing in the future (including PQ and hybrid ones) the situation of mis-selecting may happen more often

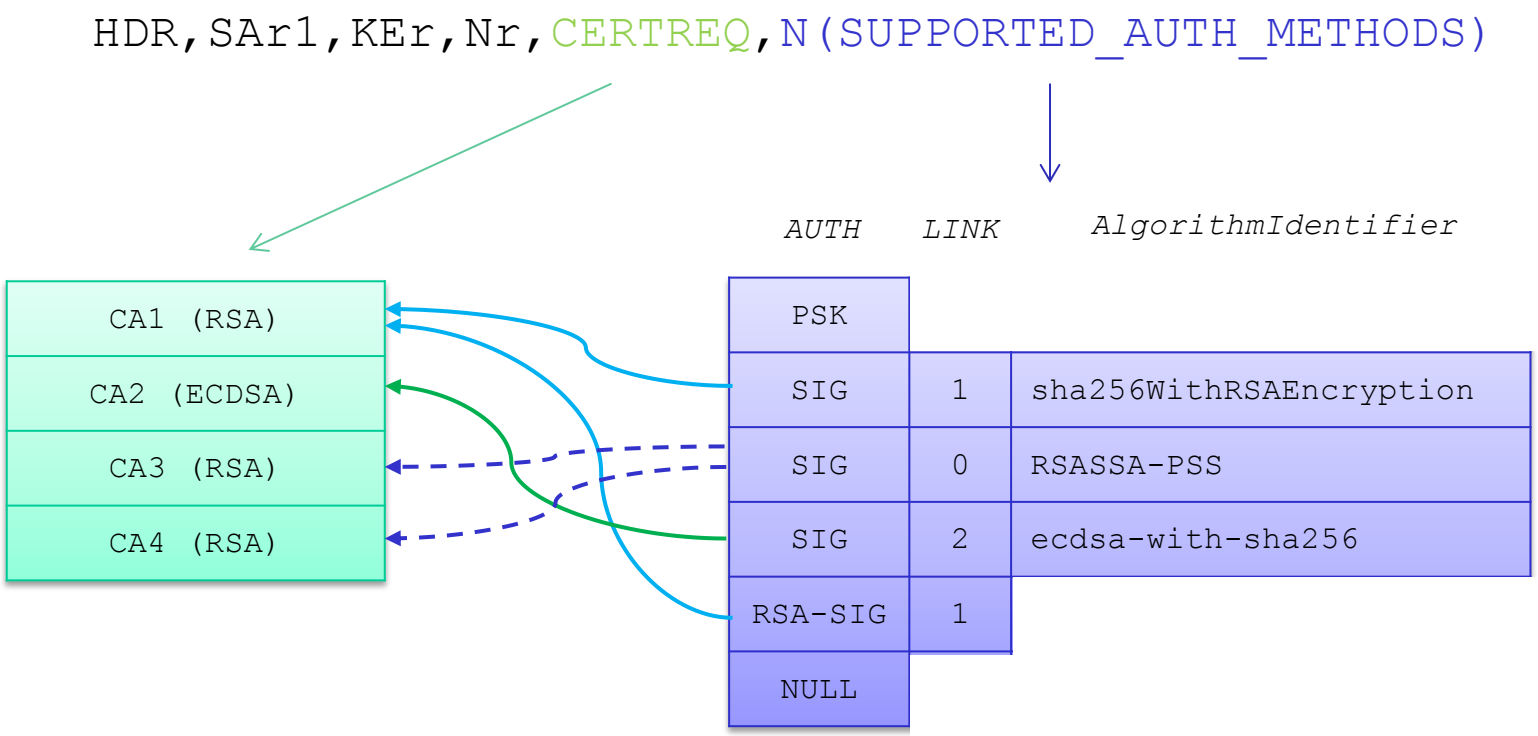
Proposed Solution

- Add new optional status notification `SUPPORTED_AUTH_METHODS` to indicate the supported authentication methods
  - for certificate-based authentication add an ability for the peers to indicate which signing algorithms can be used with each of CA in the `CERTREQ` payload
  - avoid creating new IANA registries

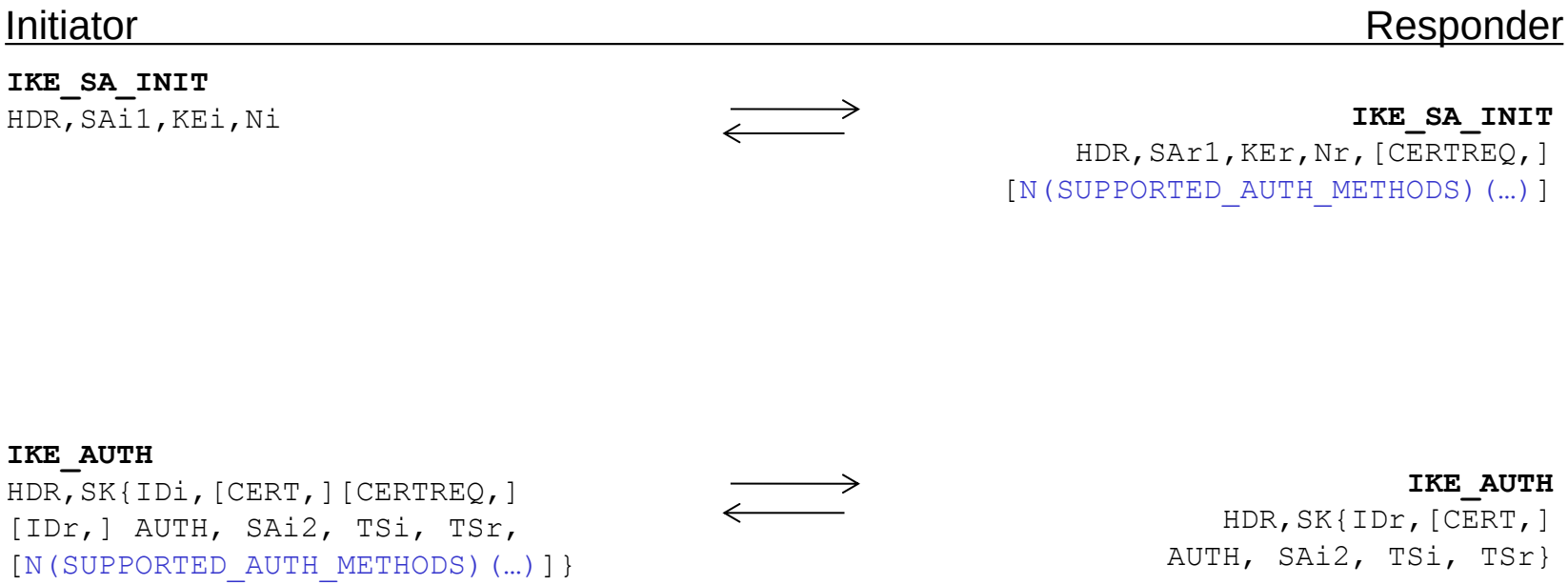
SUPPORTED_AUTH_METHODS Notification Format

- Notification data consists of a list of supported authentication methods in the following formats:
  1. Two-octet format for the methods that are not linked to `CERTREQ` payload (`PSK`, `NULL`)
  2. Three-octet format that allows optional linking to `CERTREQ` payload (`RSA-SIG` etc.)
  3. Multi-octet format that allows optional linking to `CERTREQ` payload and specifying ASN.1 `AlgorithmIdentifier` for use with particular CA (`SIG`)
- The linking to CAs is done by specifying the ordinal number of CA within the `CERTREQ` payload the method can be used with

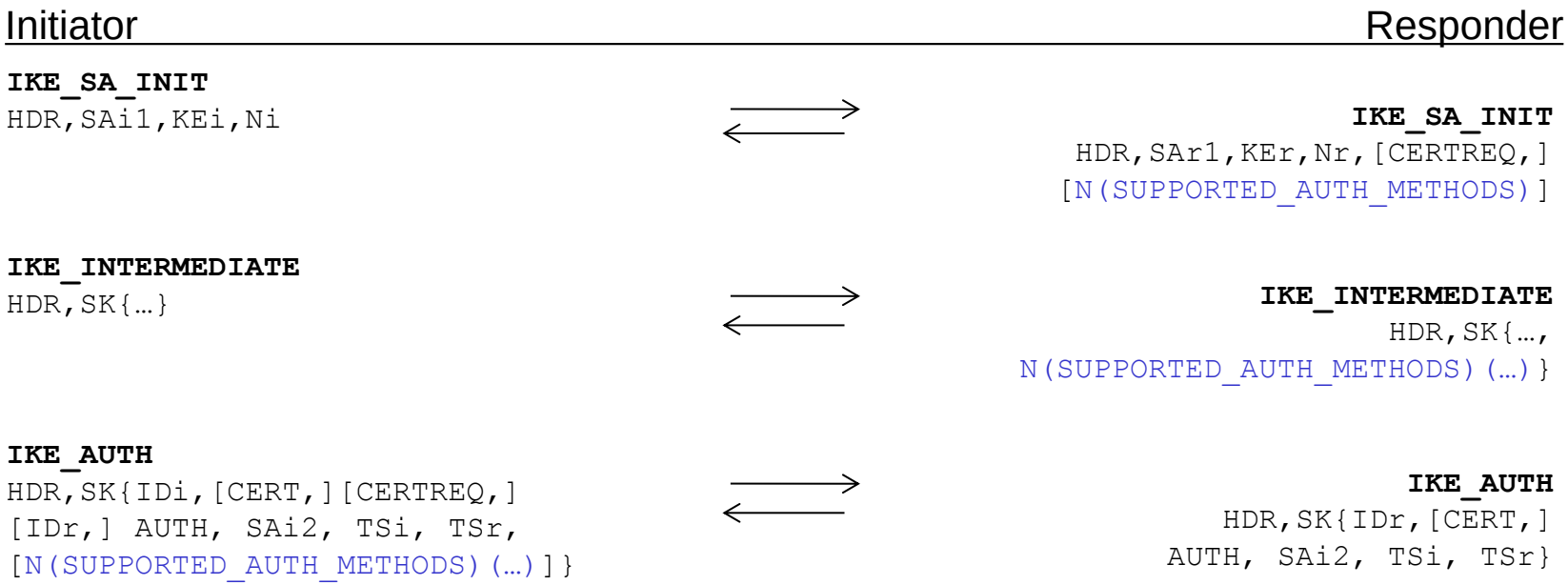
SUPPORTED_AUTH_METHODS Notification Format Illustration



Exchanges (Option 1)



Exchanges (Option 2)



Thanks

- Comments? Questions?
- More details in the draft
- WG adoption?

Open Discussion

- Other points of interest?