A Formal analysis of EDHOC Key Establishment

https://arxiv.org/abs/2007.11427

Progress report

Karl Norrman
Ericsson Research/KTH Royal Institute of Technology
2020-07-31

Joint work

- Karl Norrman, Ericsson Research / KTH Royal Institute of Technology
- Vaishnavi Sundararajan, Ericsson Research
- Alessandro Bruni, IT University of Copenhagen

- This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

Outline

- Formal analysis and the Tamarin tool
- EDHOC framework
- Analysis
  - Overview
  - Attacker model
  - Session key authentication
    - Injective agreement and implicit key authentication
  - Session key material definition

Formal analysis

- Roughly:
  - Model the protocol in some logical / mathematical formalism
  - Encode the requirements and goals in logic formulas
  - Try to prove that the formulas are true for that model (preferably with the aid of a mechanized tool)

- Useful because:
  - Modeling/proving reveal hidden assumptions, inconsistencies, incomplete requirements, vulnerabilities, …
  - Forces considering attacker models and protocol goals
  - Clarifies things: what are talking about, really? Are talking about the same thing?

- What it does **NOT** do: prove that the protocol "is secure", because
  - Not possible to define "is secure" to make everyone satisfied (new threats may also appear in the future)
  - Model may not be comprehensive (missing attacker capabilities, not covering all aspects of protocol, …)
  - All properties we care about may not be known
  - Model may not capture intention of specification accurately
  - Model abstractions may lose critical details of protocol or environment

Tamarin conceptual overview

Simplified logical description of what Tamarin does:

Define protocol in terms of actions taken by participants

Define attacker capabilities (Dolev-Yao capabilities built in)

Define properties over the model traces in an LTL-style logic

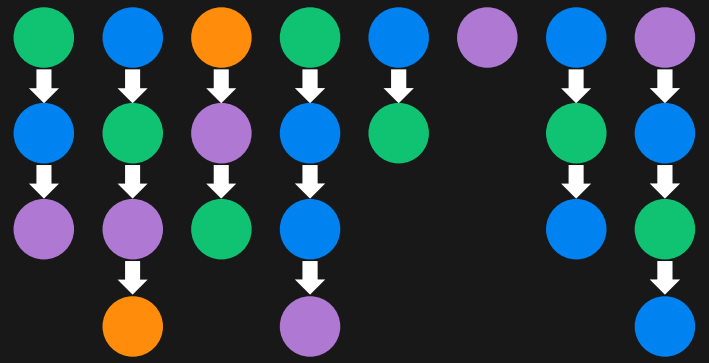
Tamarin checks whether the properties hold over the traces

Similar to a parallel functional program

Model definition

Generate

All possible execution traces of the model



Verify

LTL formulas over trace events and knowledge sets

EDHOC framework

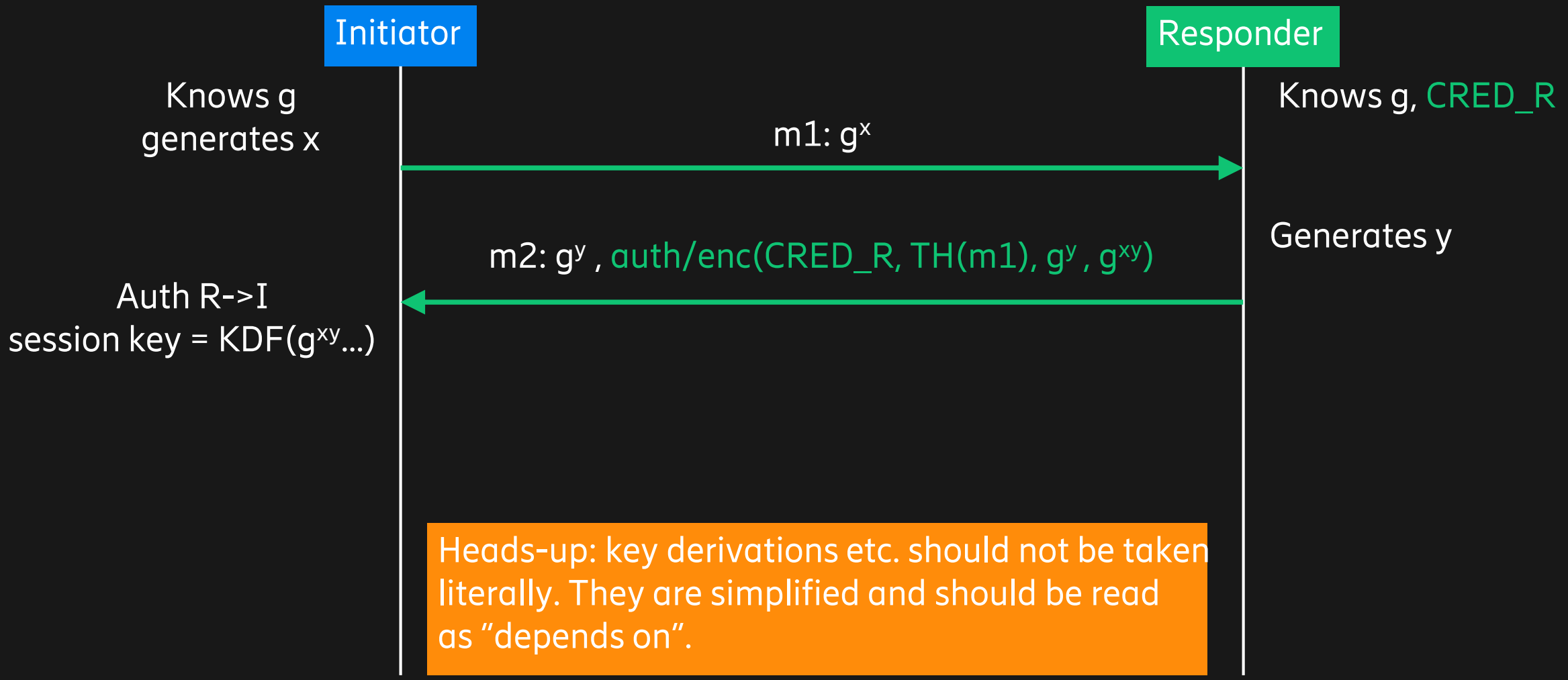
- Essentially follows the Noise framework
  - With the addition that signatures are added as authentication method (which results in a SIGMA-style cryptographic core and two mixed methods)
  - Also adds COSE and CBOR encodings, but we have not modeled that much detail

- Re-uses the challenge-response signatures for STAT based methods to reduce message size, TH (like TLS etc), g^{xy} as session key (potentially adding g^{Iy} and/or g^{Rx} in the style of OPTLS).

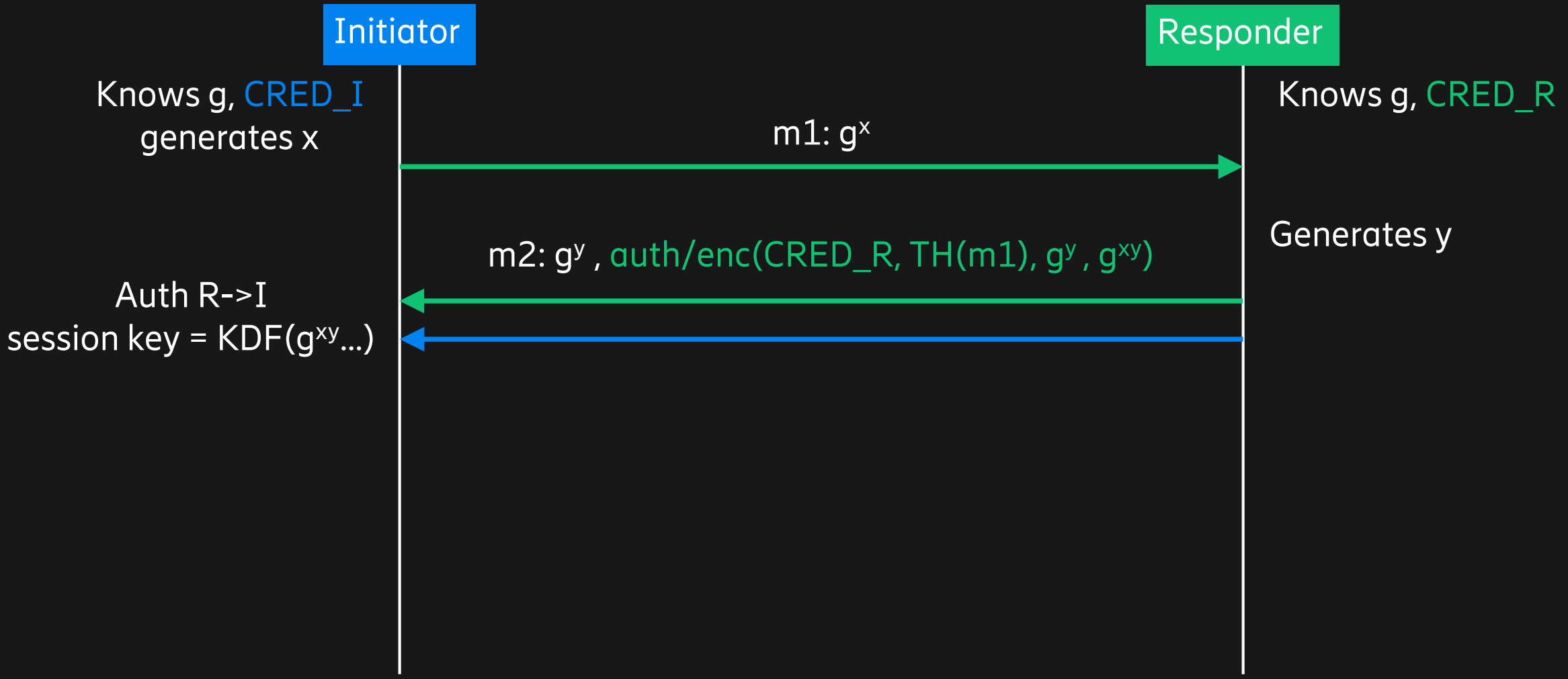
- Unclear if there is an *exact* mapping to Noise, so proofs of those does not necessarily automatically carry over.
- However, the cryptographic cores are essentially the same and use well-understood constructions.

- Main point of interest are the mixed methods: STAT-SIG and SIG-STAT.
  - (We modelled all methods anyway)

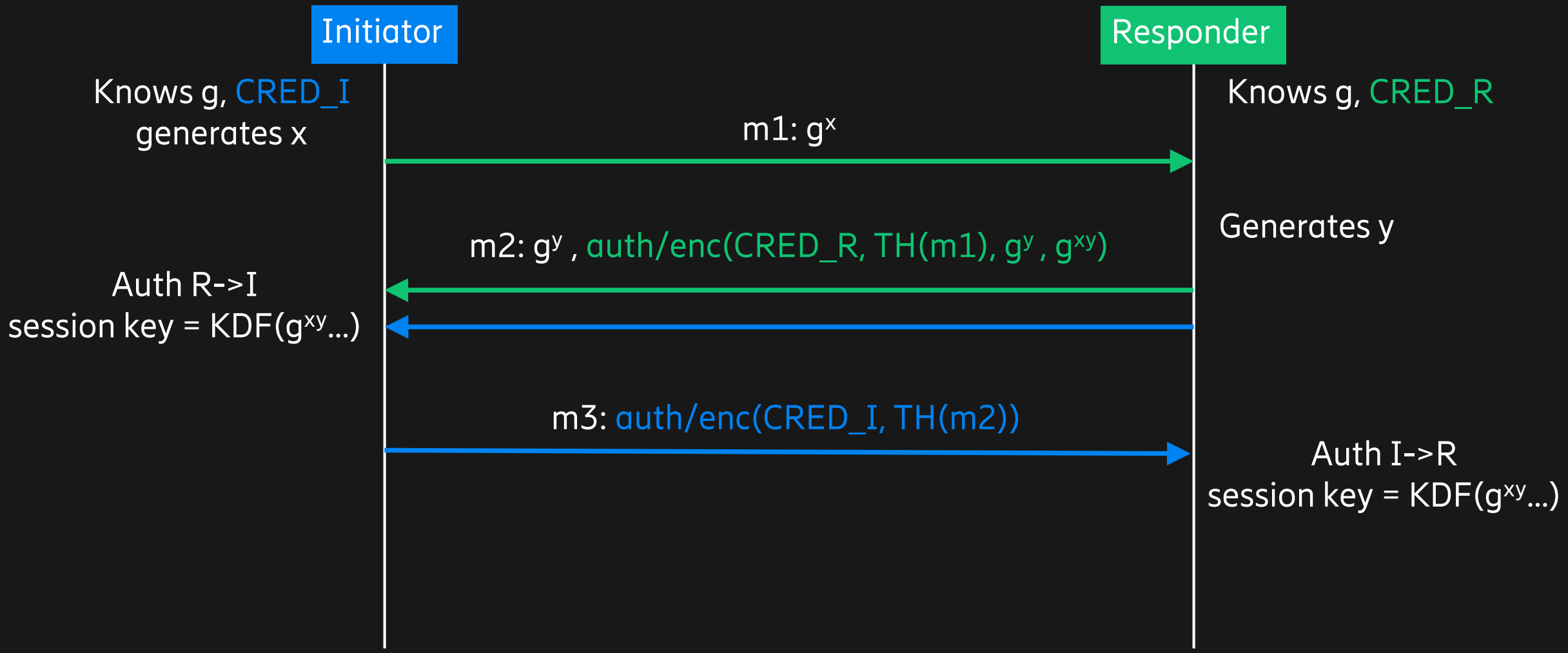
EDHOC framework – Abstract structure



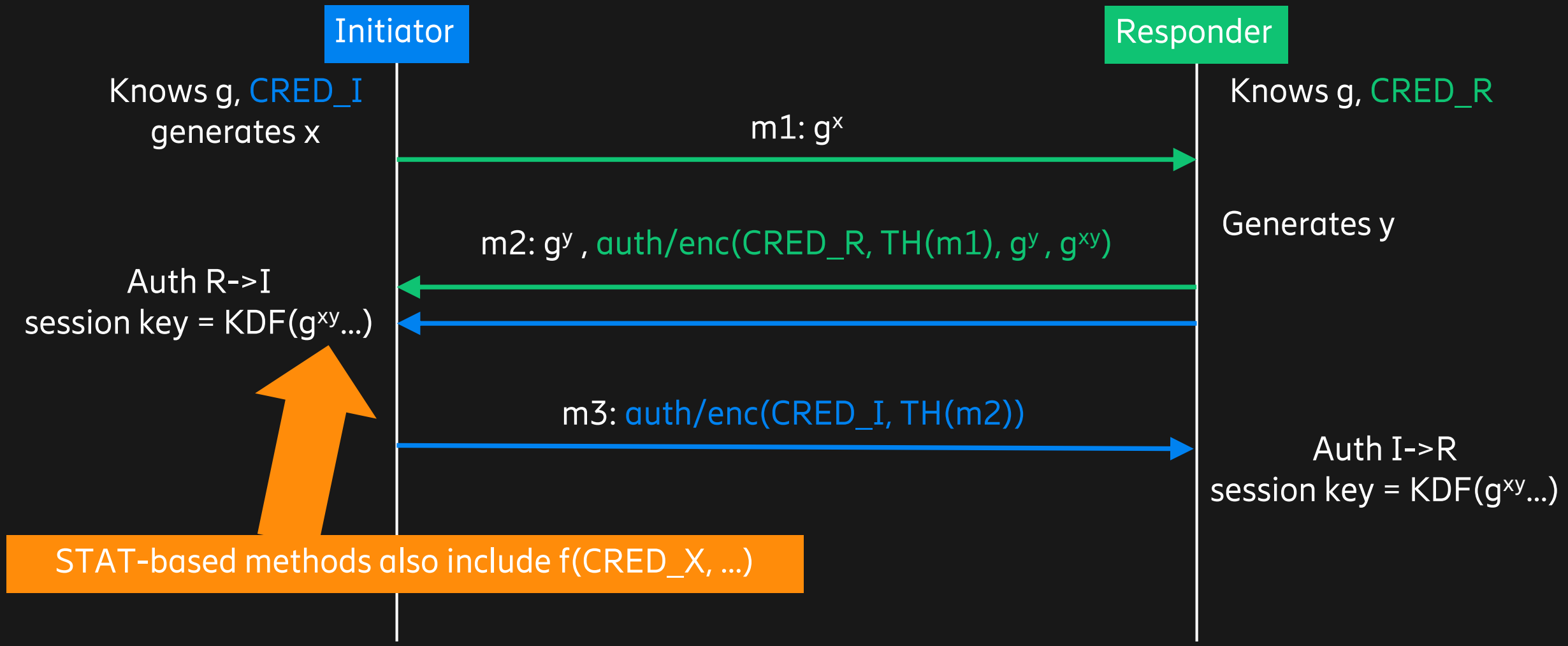
EDHOC framework – Abstract structure



EDHOC framework – Abstract structure



EDHOC framework – Abstract structure

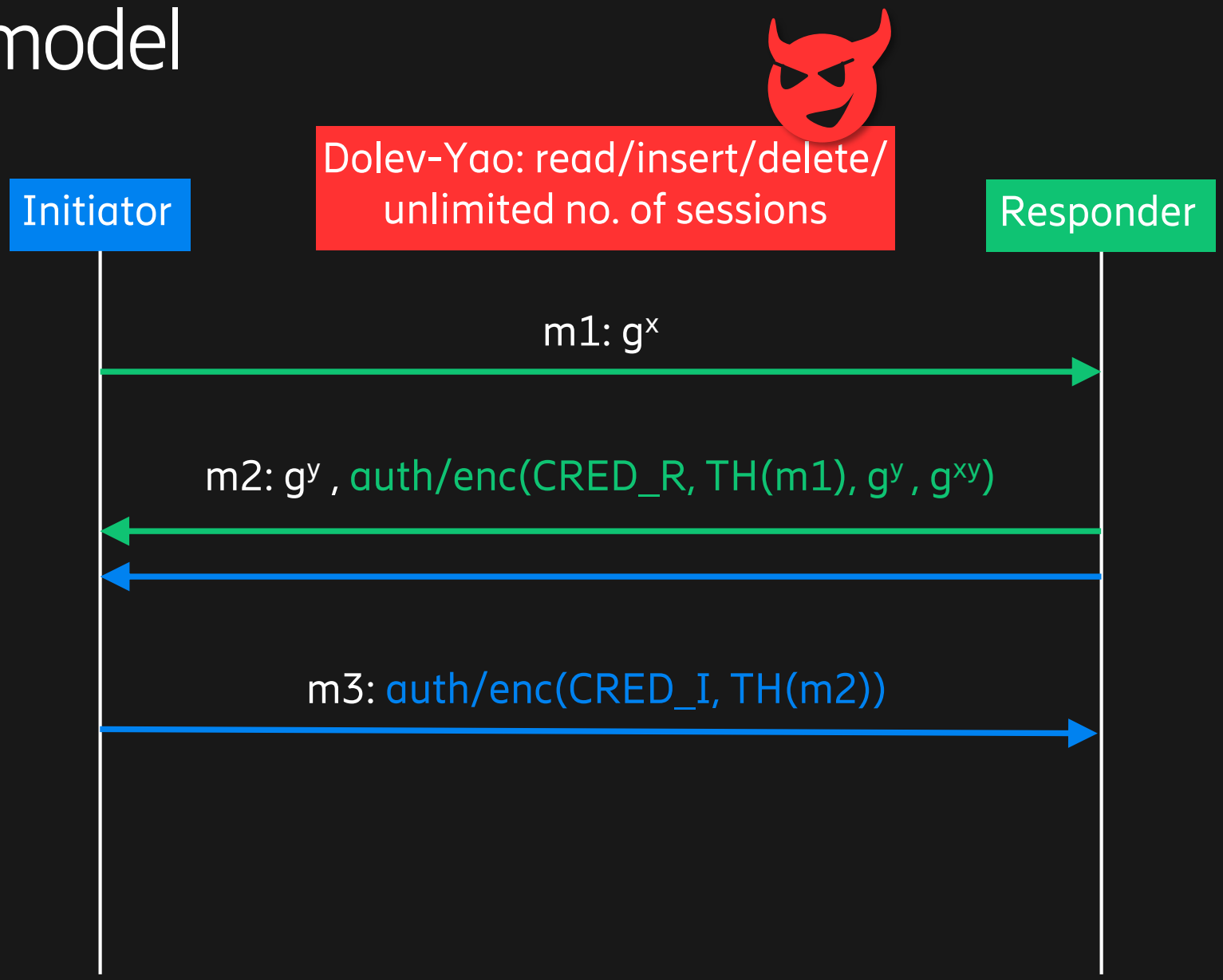


Formal model and analysis – Summary

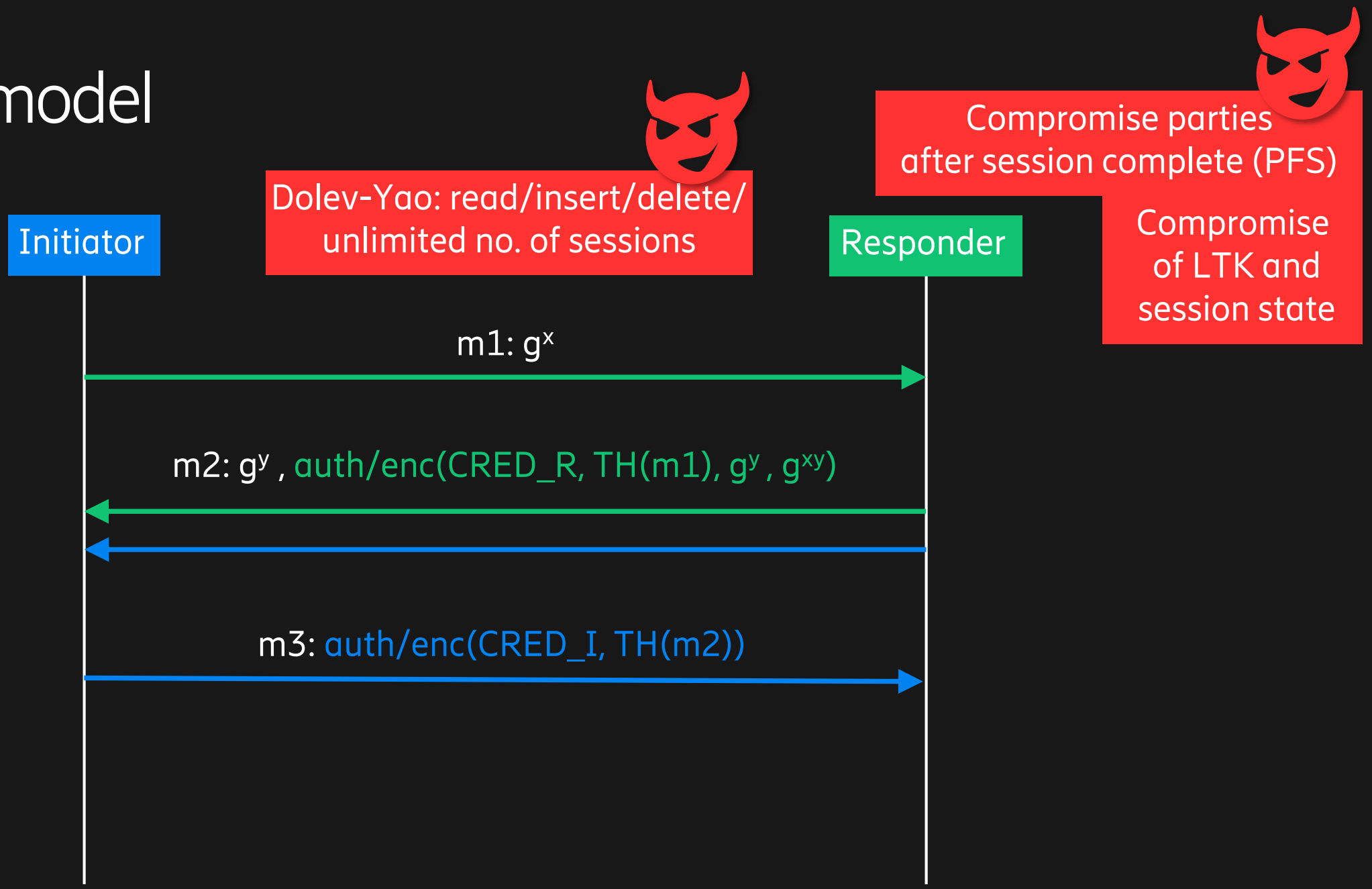
- We modeled all 5 methods in Tamarin
- We have proved fundamental properties (authentication, PFS, session key independence, …)
- Planning to prove weak-PCS (adversary gets temporary access to TPM; session keys established after access is revoked still secure) and KCI resistance
- Identified missing considerations in draft, e.g., non-repudiation aspects, taking advantage of TEEs, unintended authentication confusion, unclear intentions regarding session key definition.

- General conclusions:
  - EDHOC builds on well-established components and seems to behave as expected.
  - The different methods provide different guarantees, dictated by the credential used and the goal to keep messages small.
  - Because the EDHOC's goal is to establish a general purpose OSCORE security context, it is not clear which properties are most important to verify.
    - Trade-offs are needed as usual, exploring user-stories or use-cases would help identify which trade-offs to make (see session key authentication in the following).

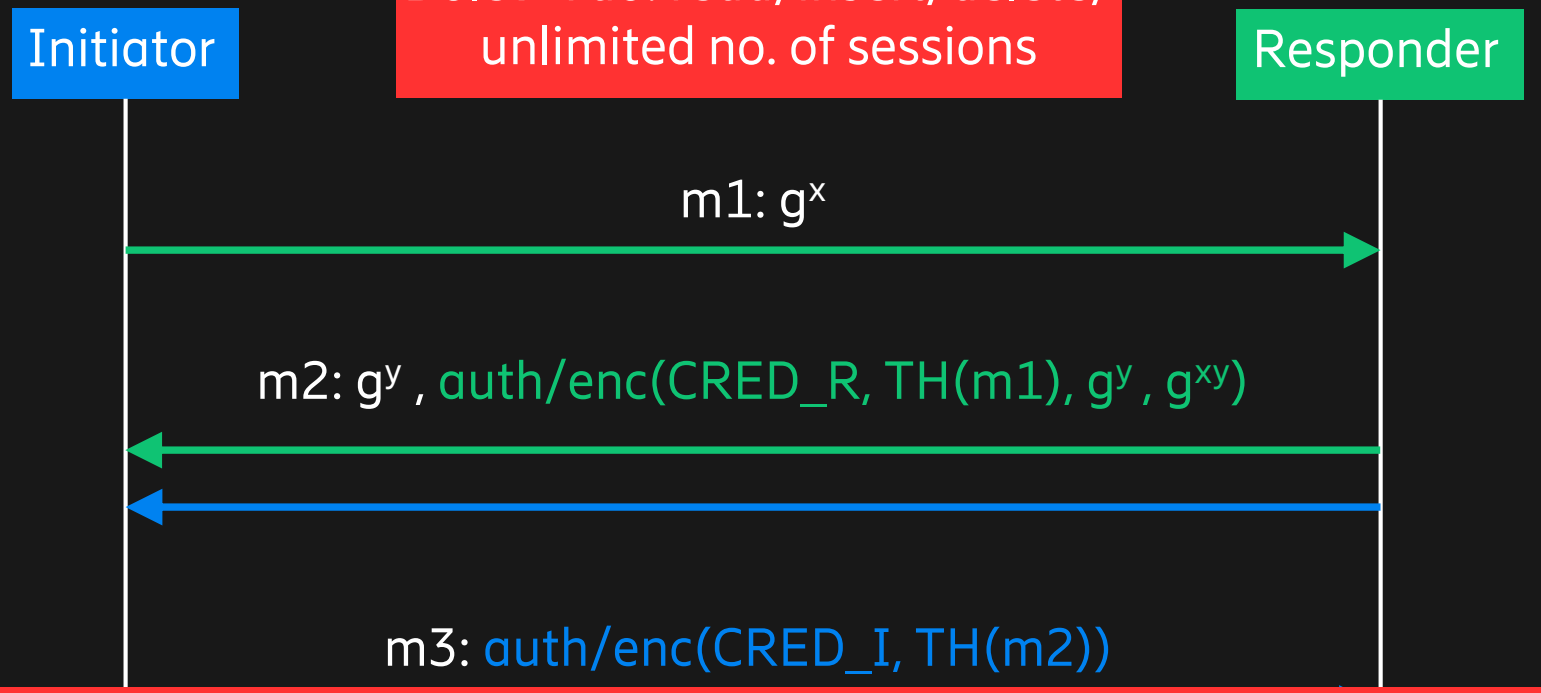
Attacker model



Attacker model



Attacker model



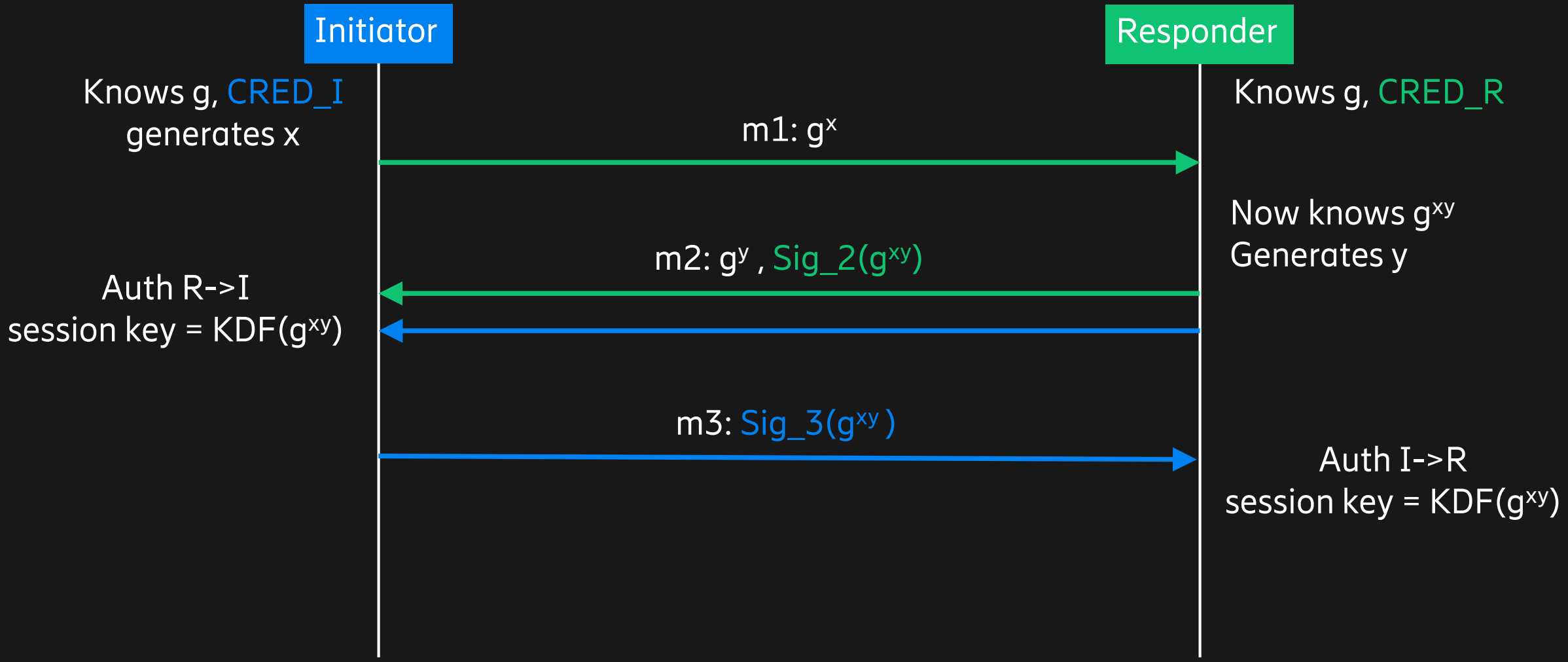
Dolev-Yao: read/insert/delete/ unlimited no. of sessions

Compromise parties after session complete (PFS)

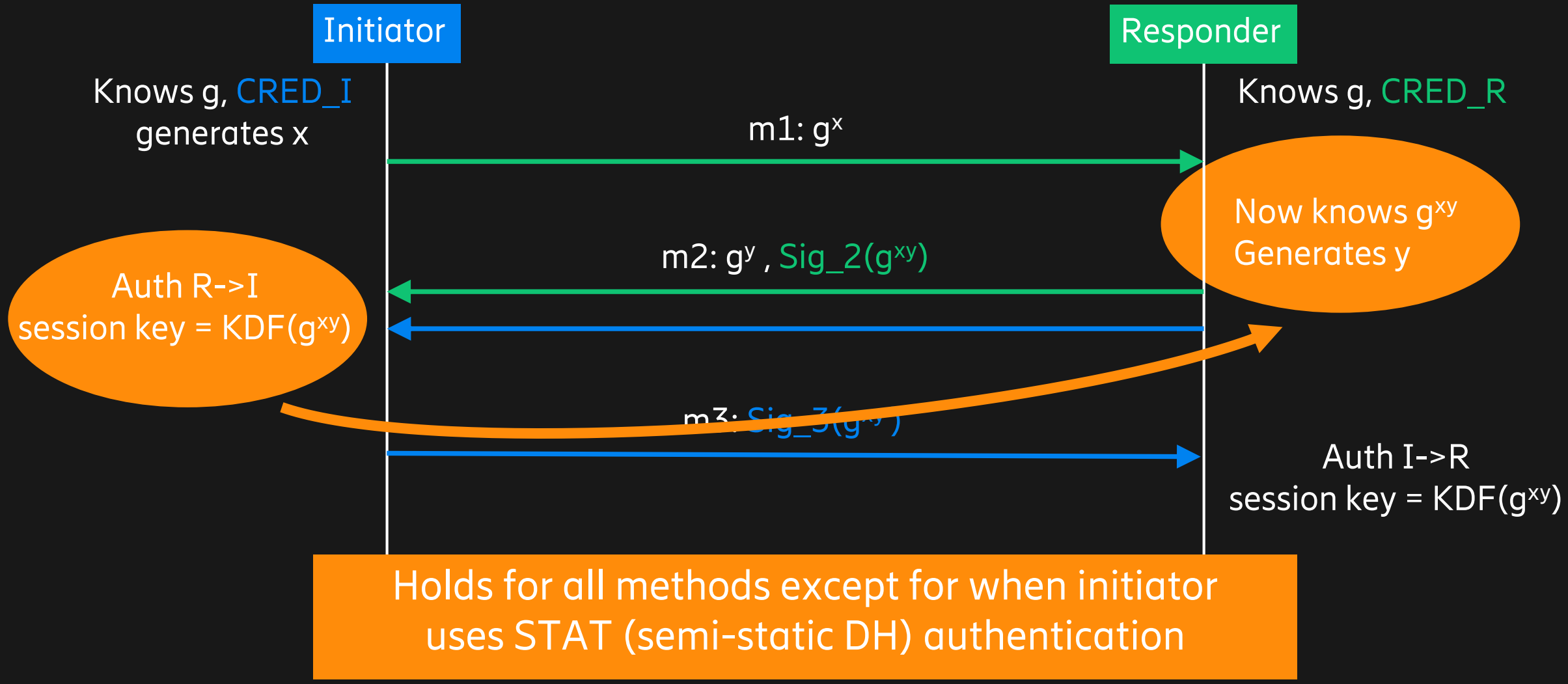
Compromise of LTK and session state

Pre-specified peer model
Attacker cannot register new keys with existing IDs in PKI (but can compromise parties any time)
Symbolic derivability based notion of secrecy
Authentication as correspondence properties over traces
Non-compromised parties are honest

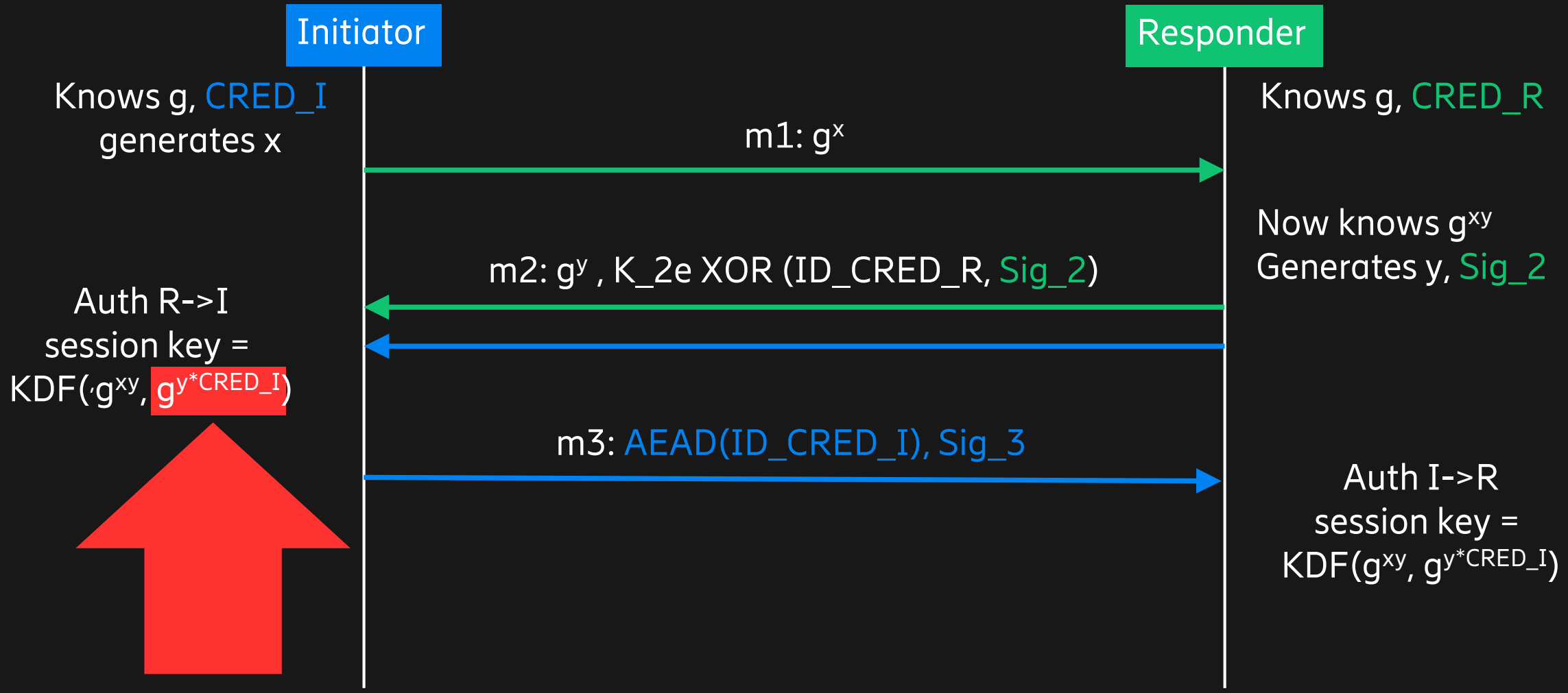
Session key authentication (injective agreement)
SIG-SIG (much simplified)



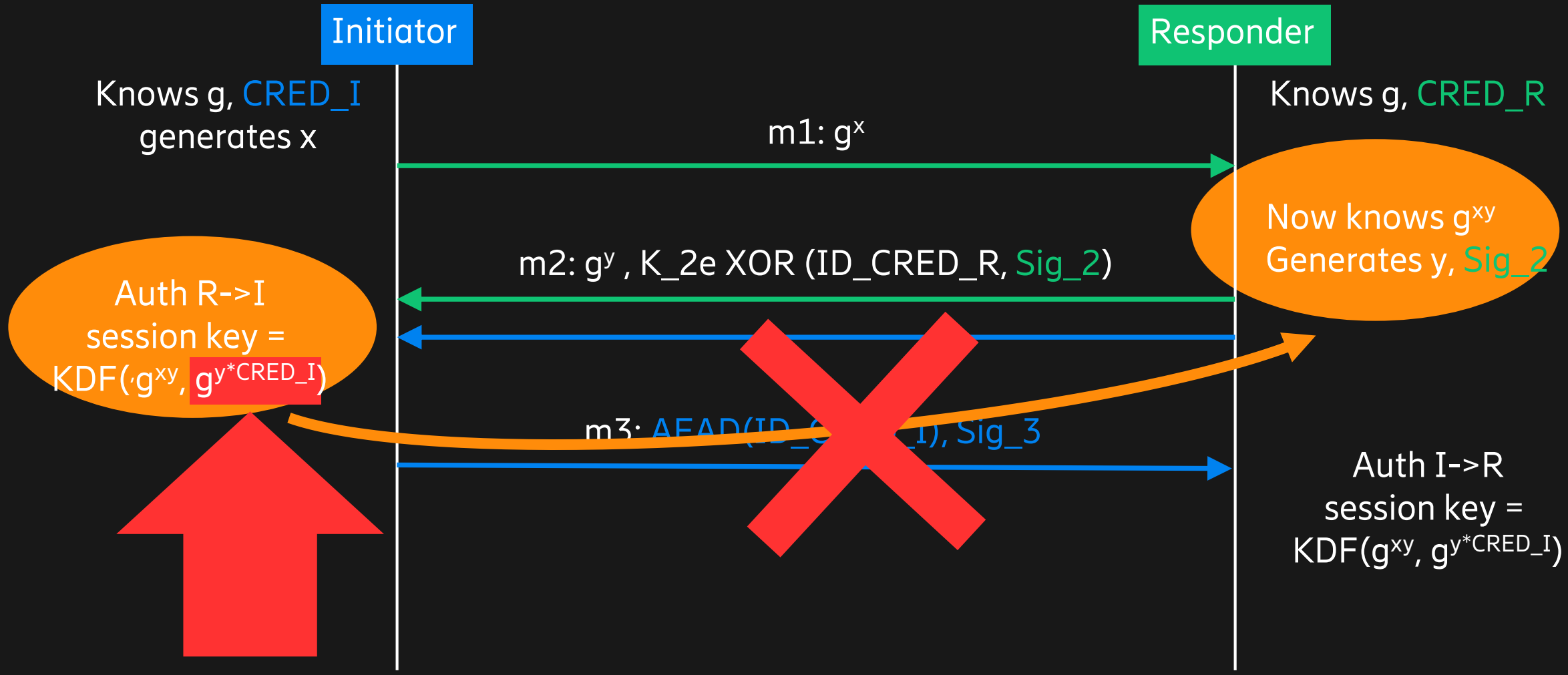
Session key authentication (injective agreement)
SIG-SIG (much simplified)



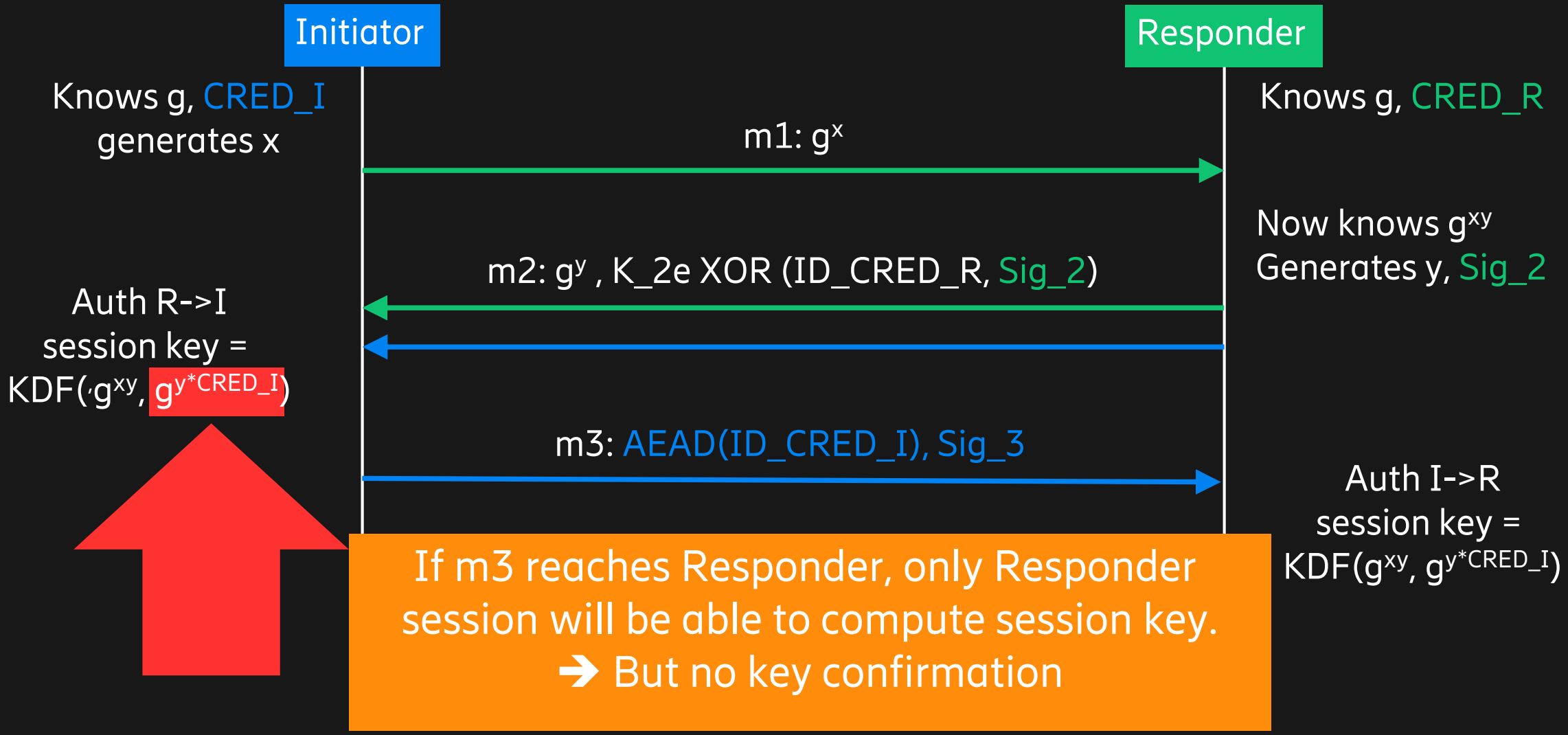
Session key authentication (injective agreement)
STAT-X (much simplified)



Session key authentication (injective agreement)
STAT-X (much simplified)



Session key authentication (implicit key authentication)
STAT-X (much simplified)



Session key material alternatives for STAT-X

1. Include semi-static key g^{y*CRED_I} and accept different properties for different methods.
2. Exclude g^{y*CRED_I}, but then differing from OPTLS (and would not take advantage of OPTLS's careful design for TEEs). PFS still holds for session key in our attacker model though, but not in a CK-style model with session state reveal queries.
3. Add a fourth message from R to I including a MAC based on key derived from key material from independent branch in key hierarchy (to not destroy key indistinguishability).
4. Include Initiator ID in message 1: Removes identity hiding of initiator.

- Without better understanding the protocol goals, selecting alternatives
- Whatever the choice: should decision be aligned across methods?

Thanks for listening!

Questions are welcome now or via mail

karl.norrman@ericsson.com