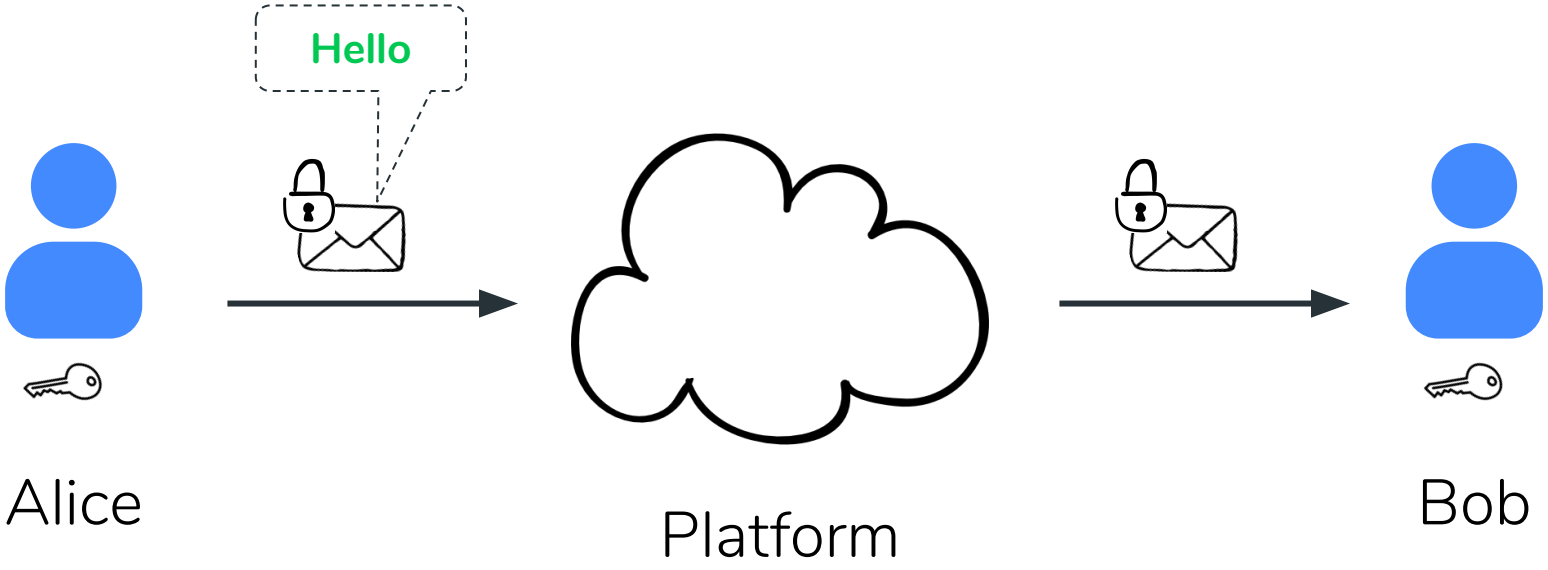
Traceback for End-to-End Encrypted Messaging

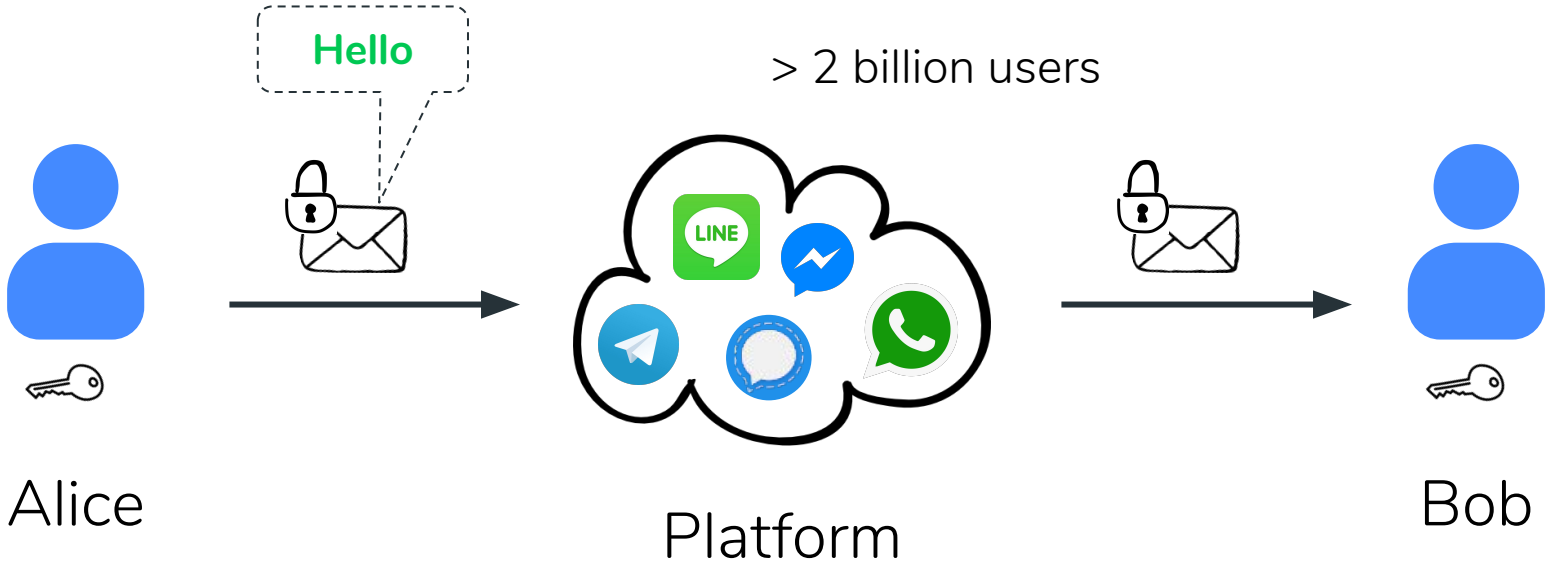
Nirvan Tyagi Ian Miers Tom Ristenpart

CCS 2019

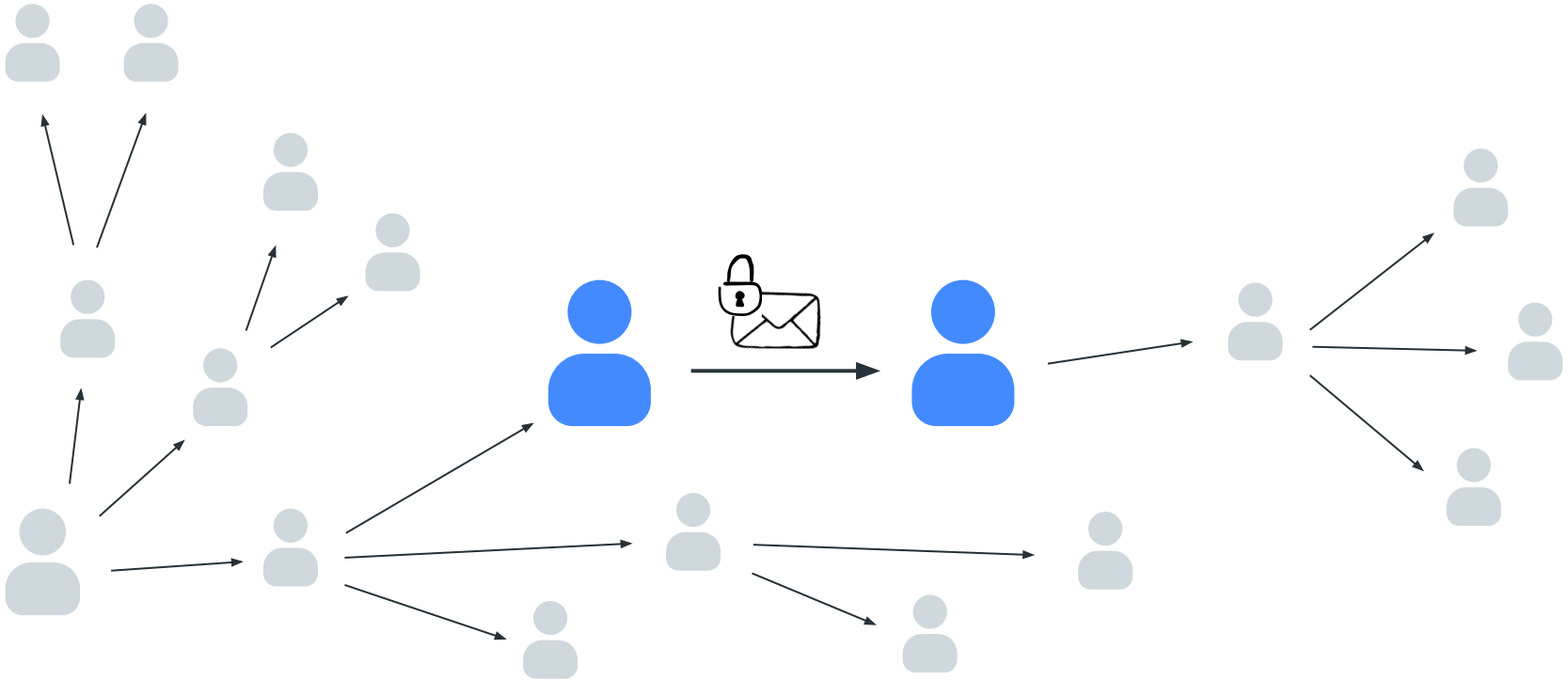
Setting: End-to-end encrypted (E2EE) messaging



Setting: End-to-end encrypted (E2EE) messaging

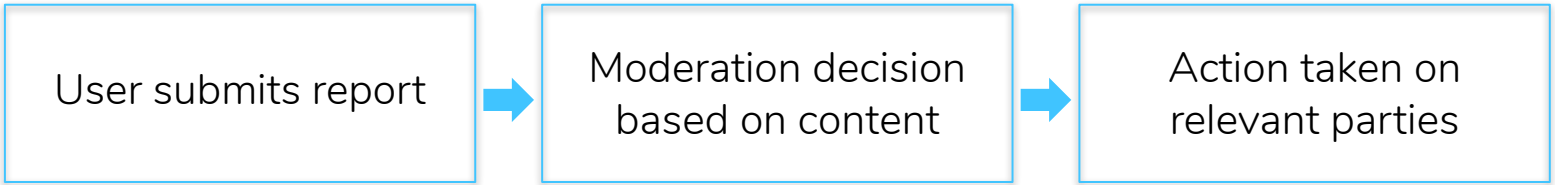


Problem: Viral forwarding of misinformation in E2EE messaging

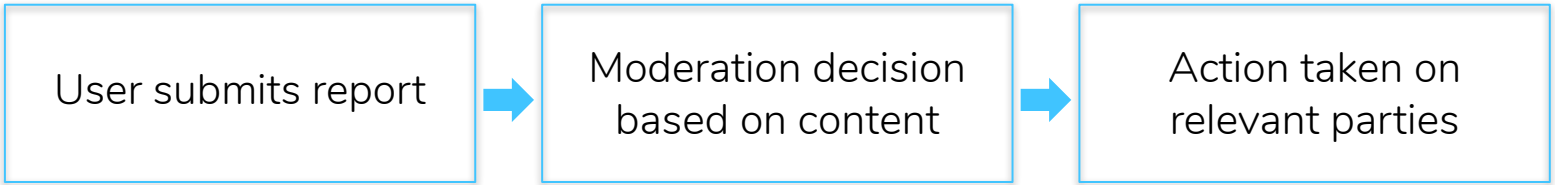


Problem: Viral forwarding of misinformation in E2EE messaging

Content moderation for user-driven reports

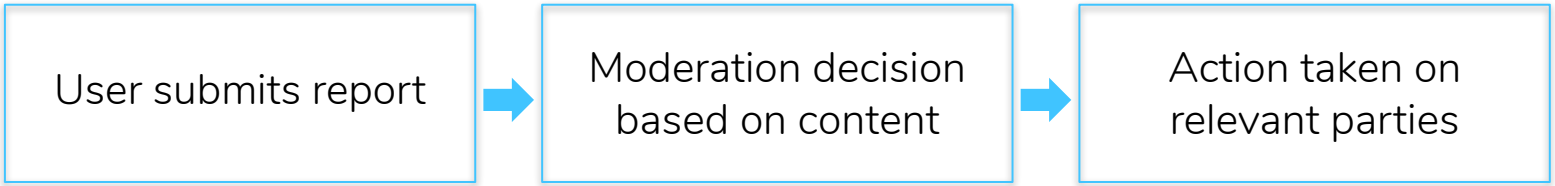


Content moderation for user-driven reports



Combination of machine learning and human review

Content moderation for user-driven reports

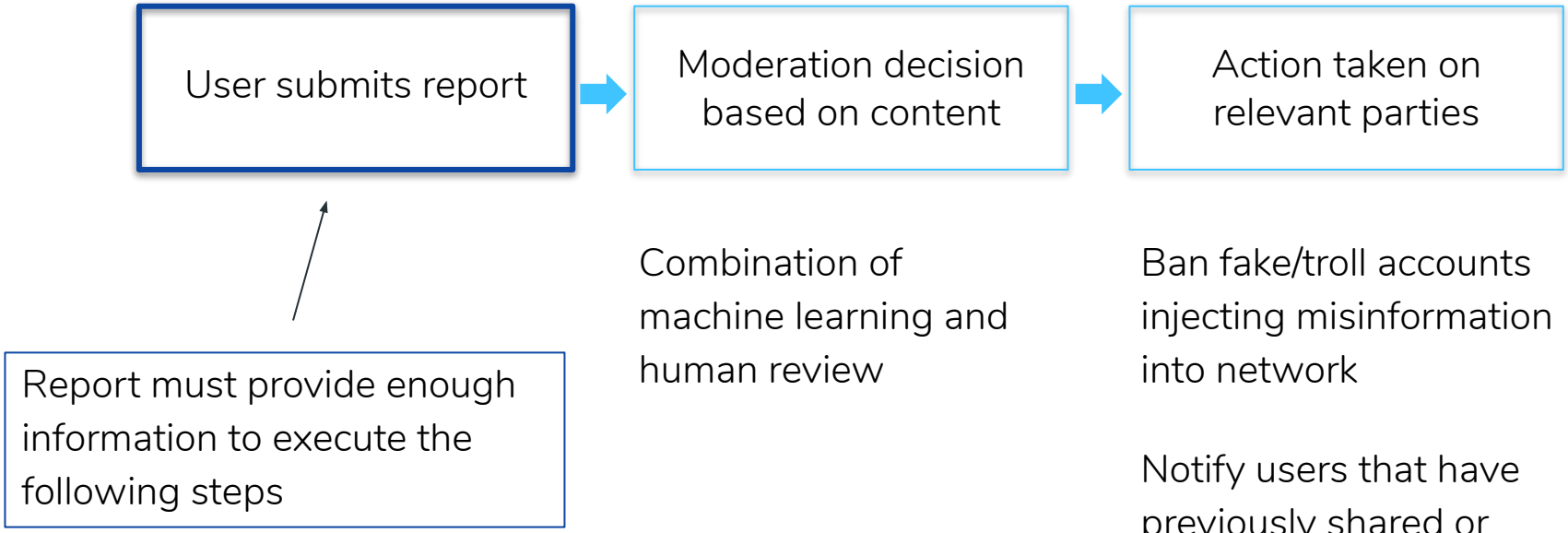


Combination of machine learning and human review

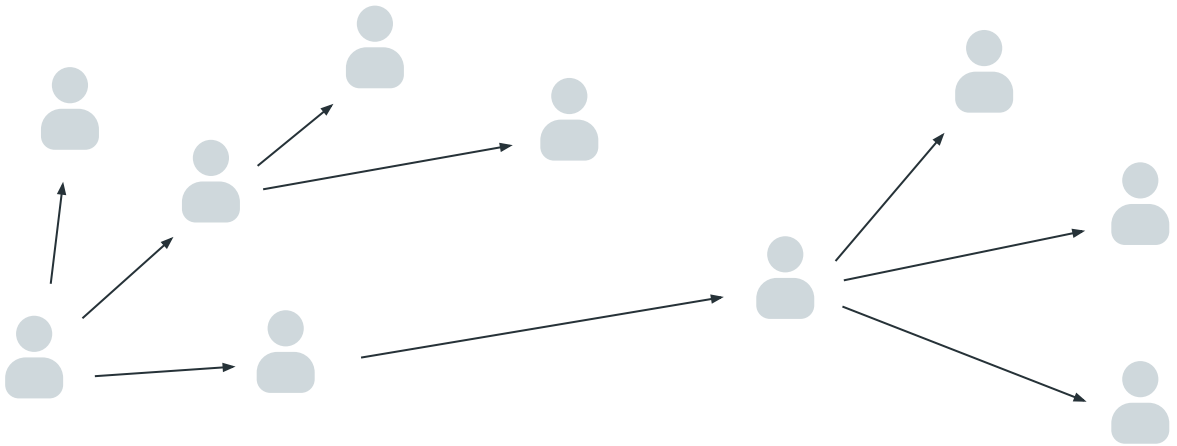
Ban fake/troll accounts injecting misinformation into network

Notify users that have previously shared or received misinformation

Content moderation for user-driven reports

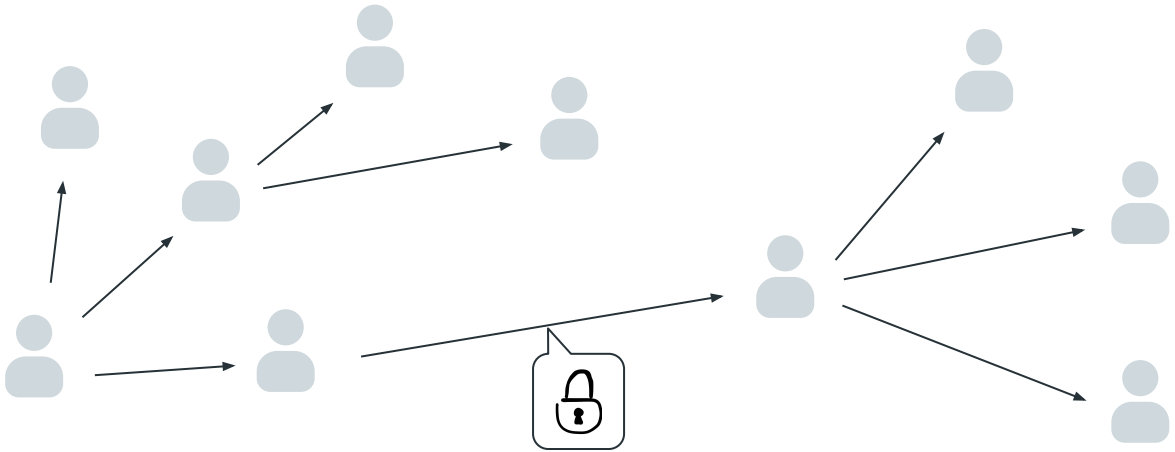


E2EE hides information useful for content moderation of misinformation



E2EE hides information useful for content moderation of misinformation

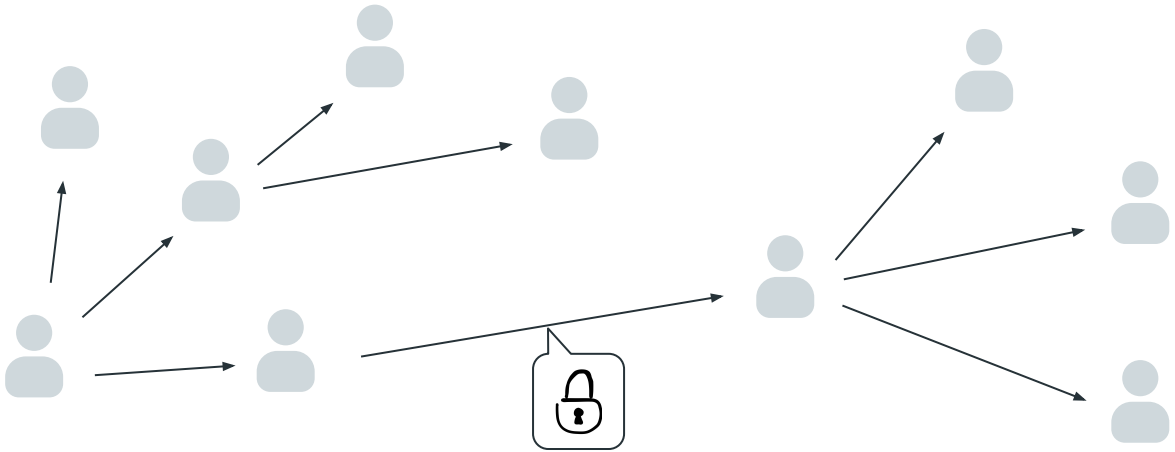
- Platform doesn’t see message content



Message content is encrypted!

E2EE hides information useful for content moderation of misinformation

- Platform doesn't see message content
- Platform doesn't see forwarding relationships

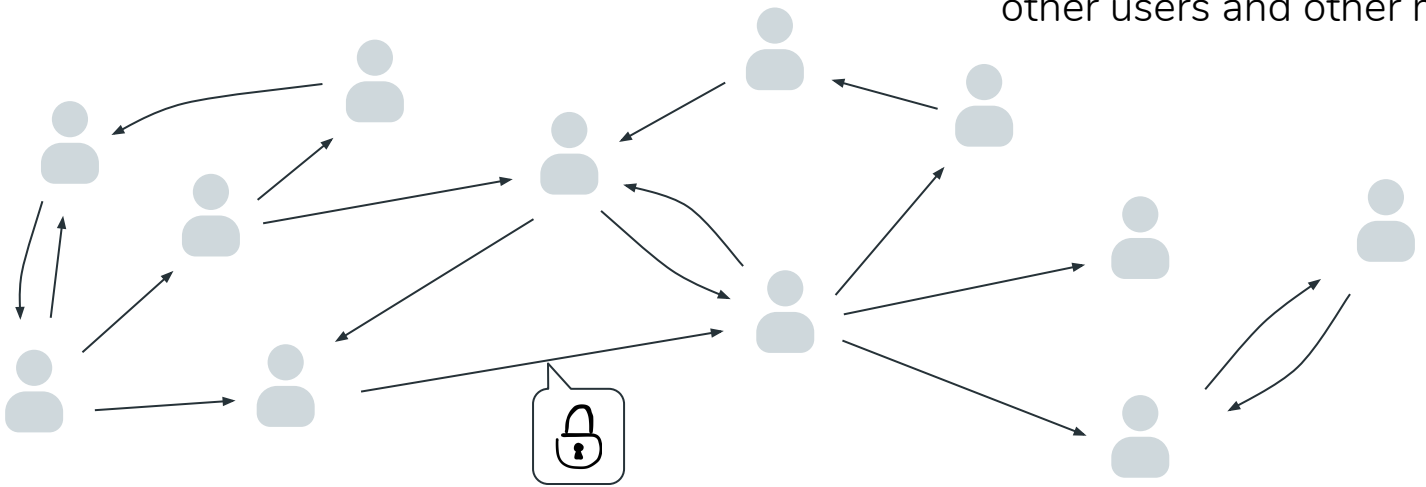


Message content is encrypted!

E2EE hides information useful for content moderation of misinformation

- Platform doesn't see message content
- Platform doesn't see forwarding relationships

Forwarding traffic is muddled by other users and other messages

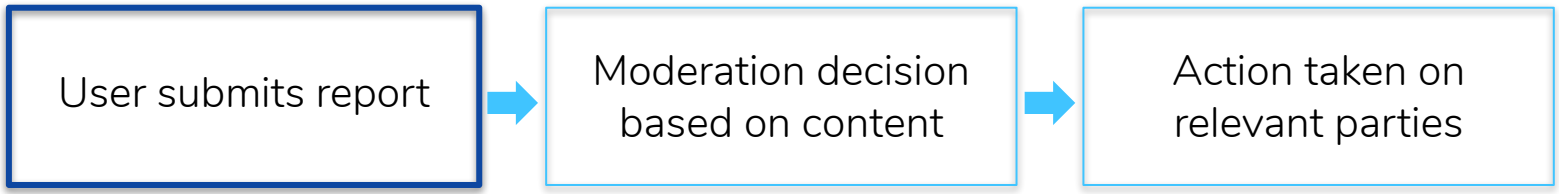


Message content is encrypted!

This work: **Tracing** in E2EE messaging

[TMR CCS'19]

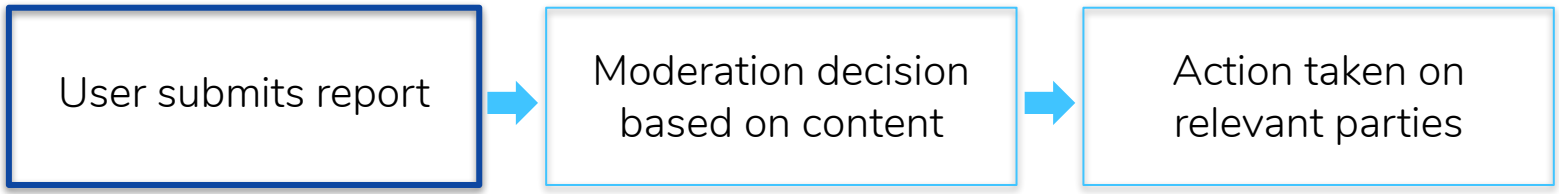
- **Message tracing**: new cryptographic functionality for user-driven reporting of forwards in E2EE messaging
  - Path traceback: chain of messages from source to reporter
  - Tree traceback: entire forwarding tree of messages originating from source



This work: **Tracing** in E2EE messaging

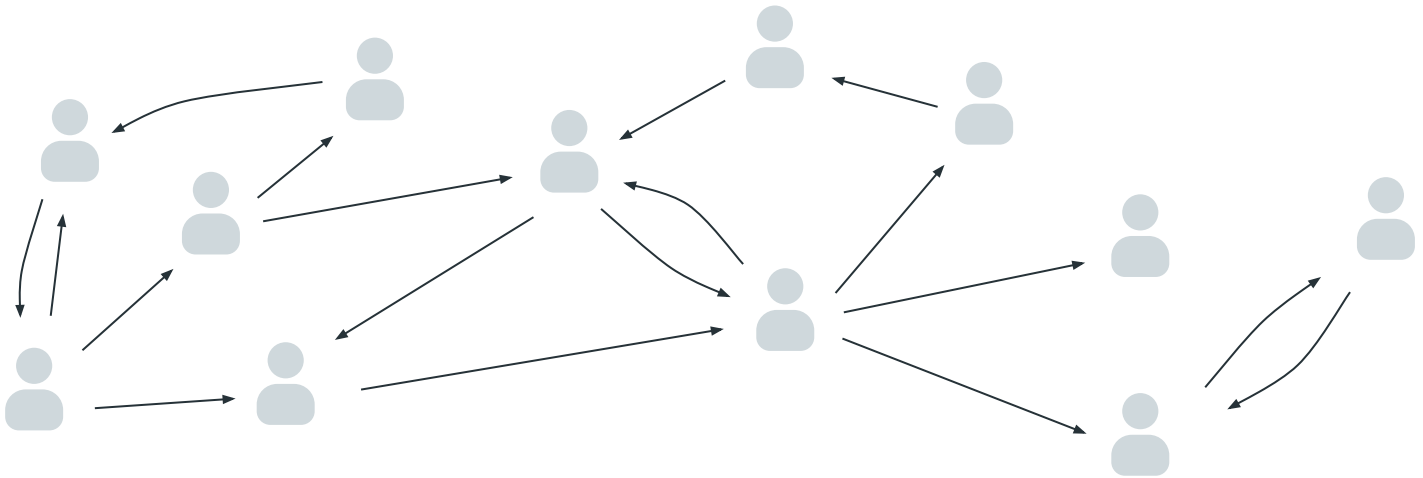
[TMR CCS'19]

- **Message tracing**: new cryptographic functionality for user-driven reporting of forwards in E2EE messaging
    - Path traceback: chain of messages from source to reporter
    - Tree traceback: entire forwarding tree of messages originating from source
- Formal confidentiality and accountability security notions for tracing
- Implementation and evaluation of practicality



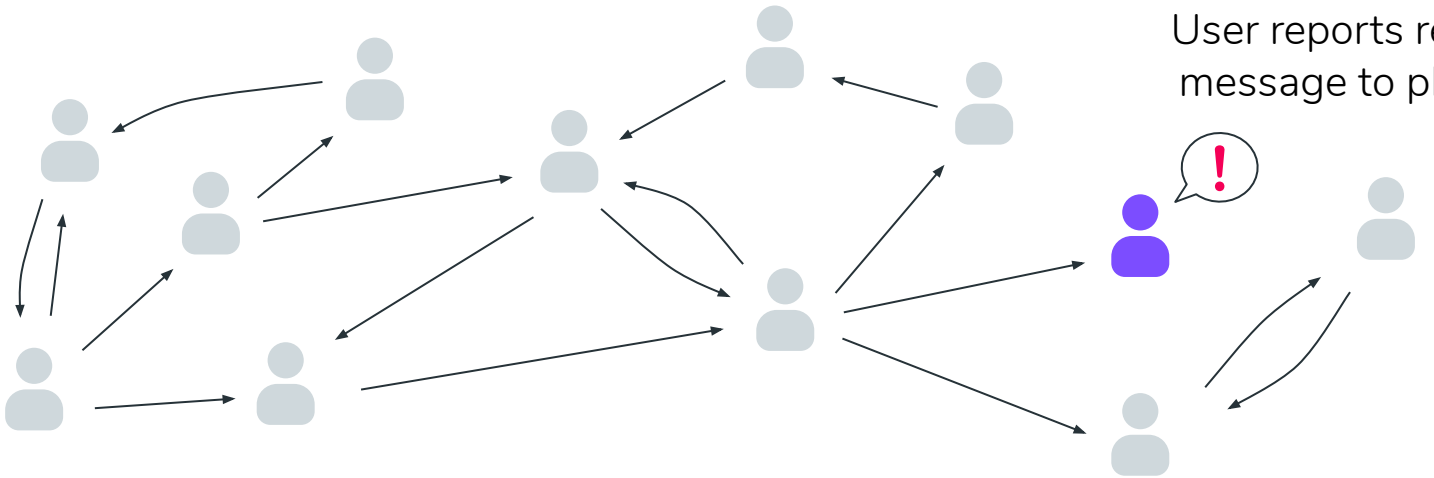
Prior work: Abuse reporting in E2EE messaging

Message franking [FB white paper ’17], [GLR CRYPTO’17], [DGRW CRYPTO’18]



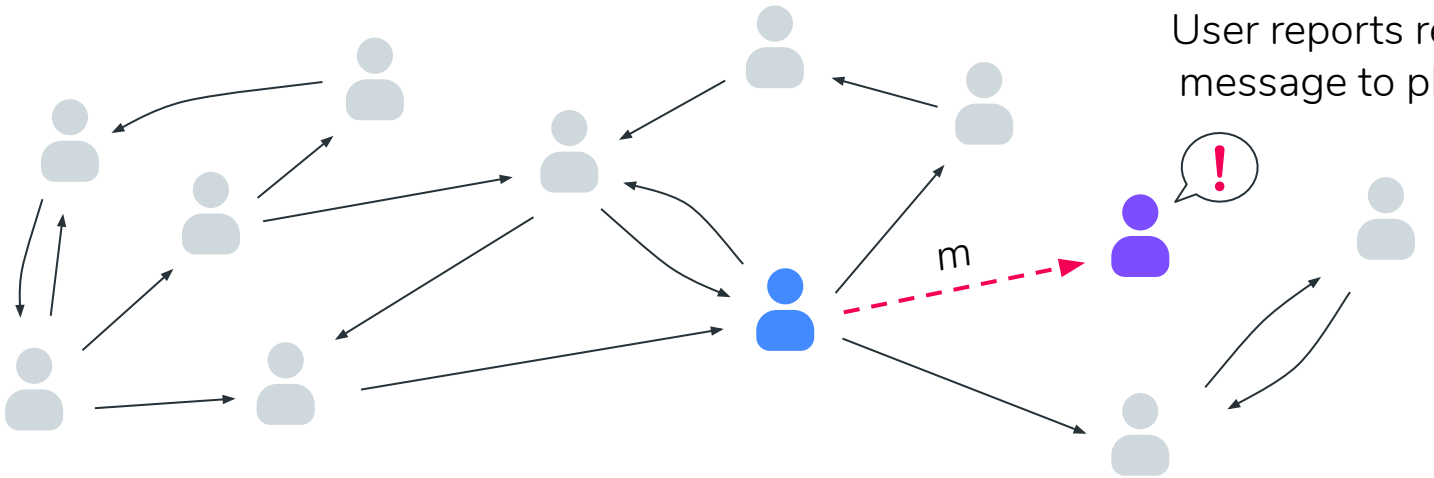
Prior work: Abuse reporting in E2EE messaging

Message franking [FB white paper ’17], [GLR CRYPTO’17], [DGRW CRYPTO’18]



Prior work: Abuse reporting in E2EE messaging

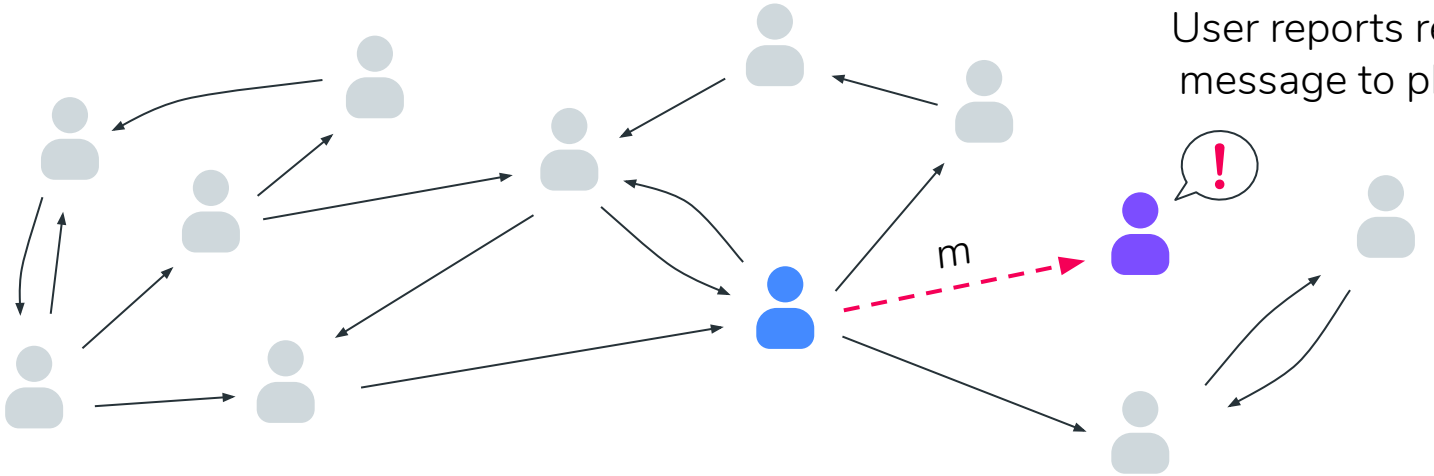
Message franking [FB white paper ’17], [GLR CRYPTO’17], [DGRW CRYPTO’18]



Prior work: Abuse reporting in E2EE messaging

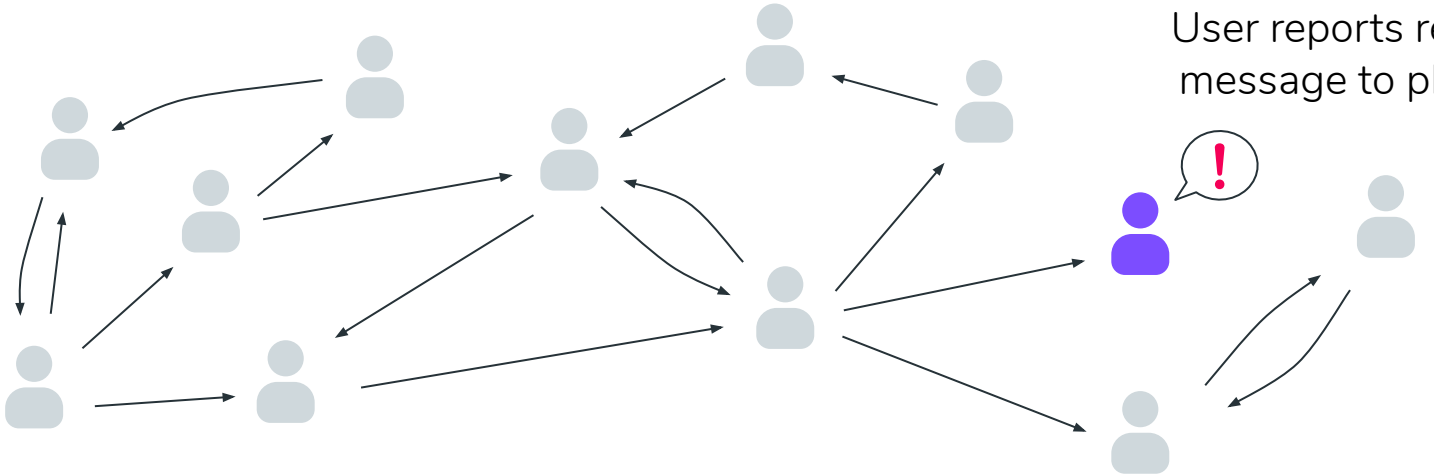
Message franking [FB white paper '17], [GLR CRYPTO'17], [DGRW CRYPTO'18]

Platform learns message and sender, but nothing more about where message came from or where it reached



This work: **Tracing** in E2EE messaging

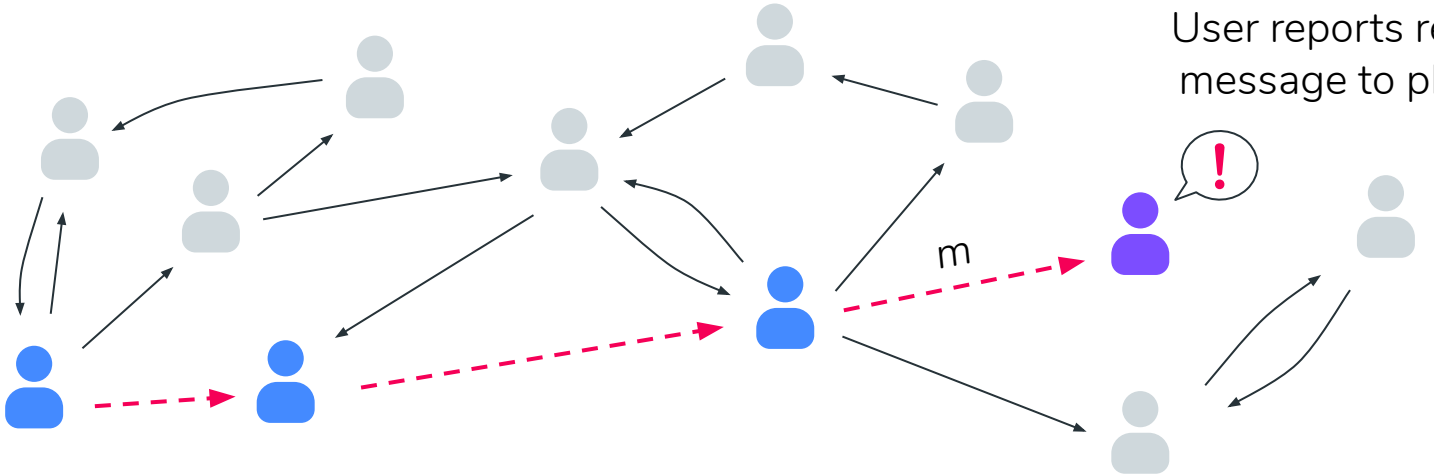
[TMR CCS'19]



This work: **Tracing** in E2EE messaging

[TMR CCS'19]

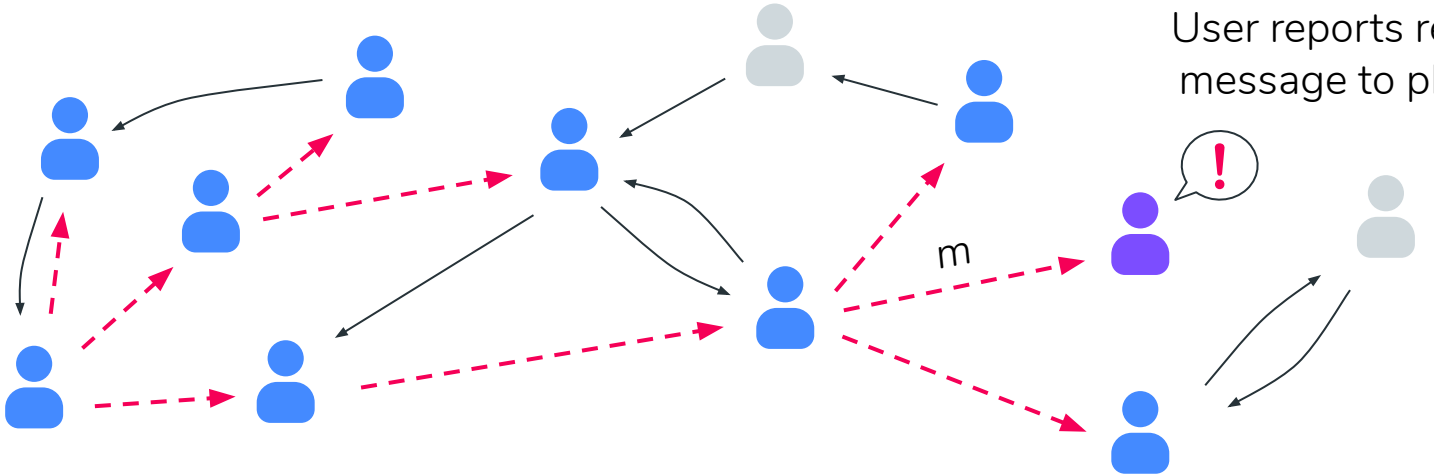
- Two constructions for message tracing
  - Path traceback



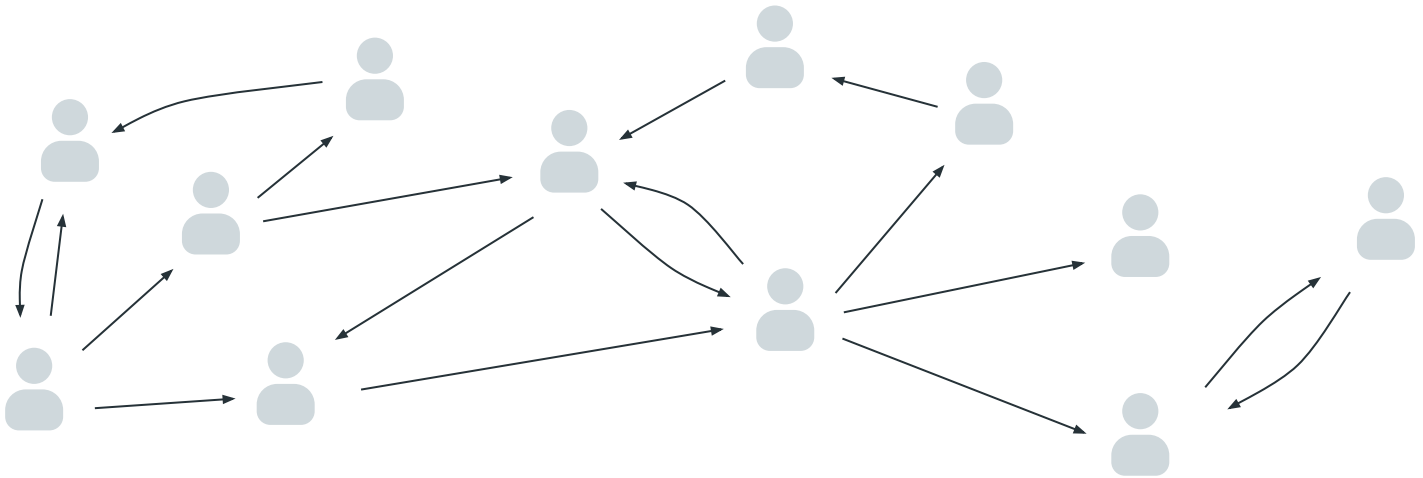
This work: **Tracing** in E2EE messaging

[TMR CCS'19]

- Two constructions for message tracing
  - Path traceback
  - Tree traceback



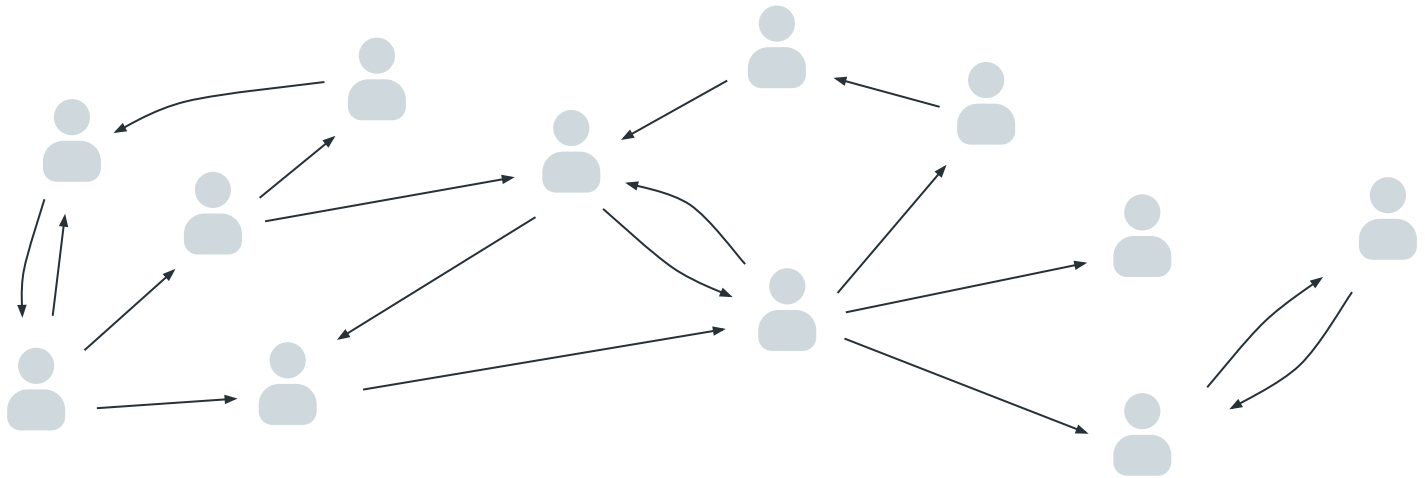
Goal: Act like standard E2EE messaging before report



Goal: Act like standard E2EE messaging before report

Before report

Platform view: encrypted content and metadata (participants, length, and timing)

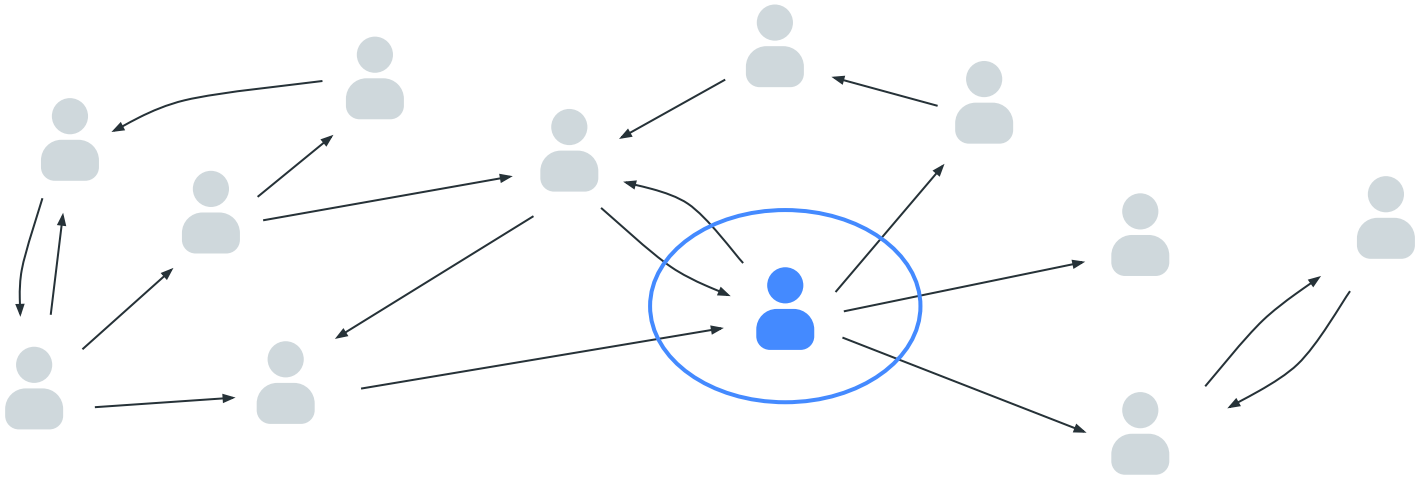


Goal: Act like standard E2EE messaging before report

Before report

Platform view: encrypted content and metadata (participants, length, and timing)

User view: messages they receive or send

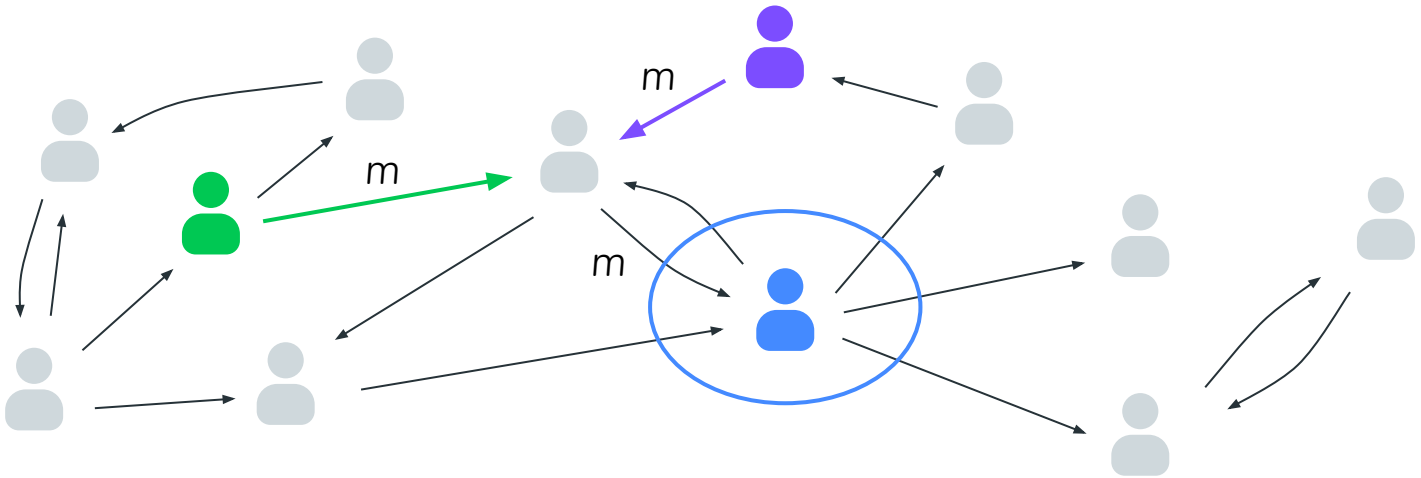


Goal: Act like standard E2EE messaging before report

Before report

Platform view: encrypted content and metadata (participants, length, and timing)

User view: messages they receive or send



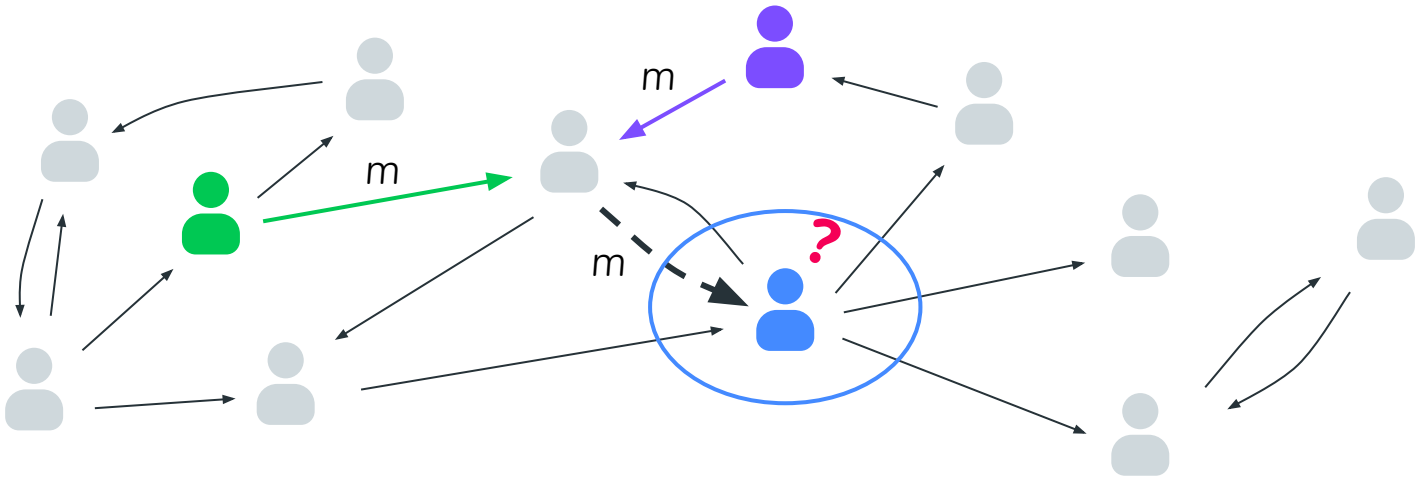
User shouldn't learn forwarding info of received messages

Goal: Act like standard E2EE messaging before report

Before report

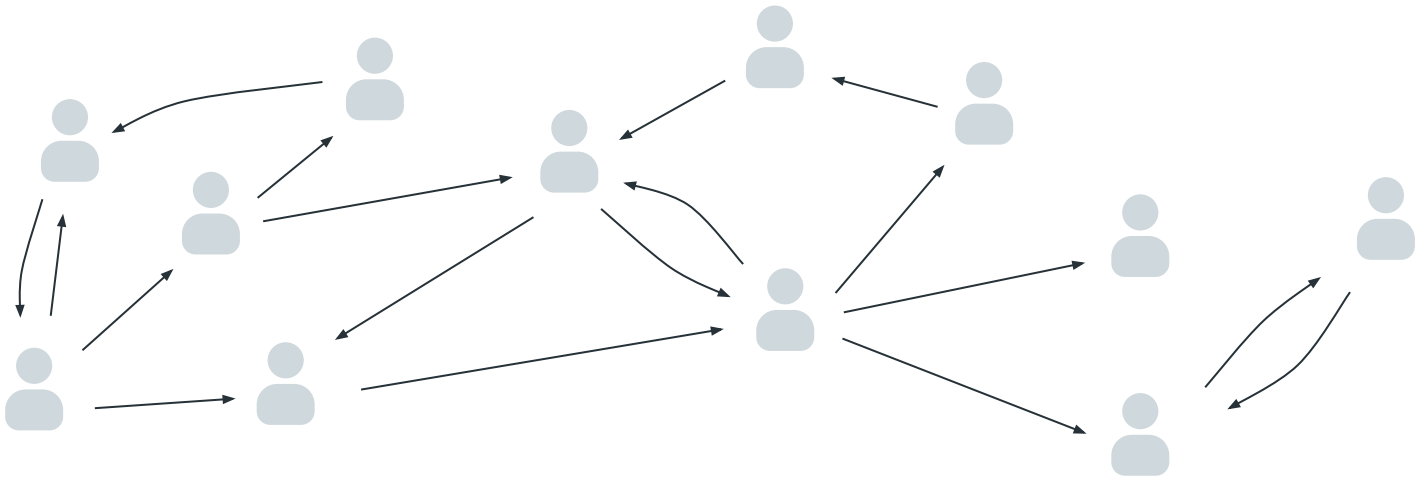
Platform view: encrypted content and metadata (participants, length, and timing)

User view: messages they receive or send



User shouldn't learn forwarding info of received messages

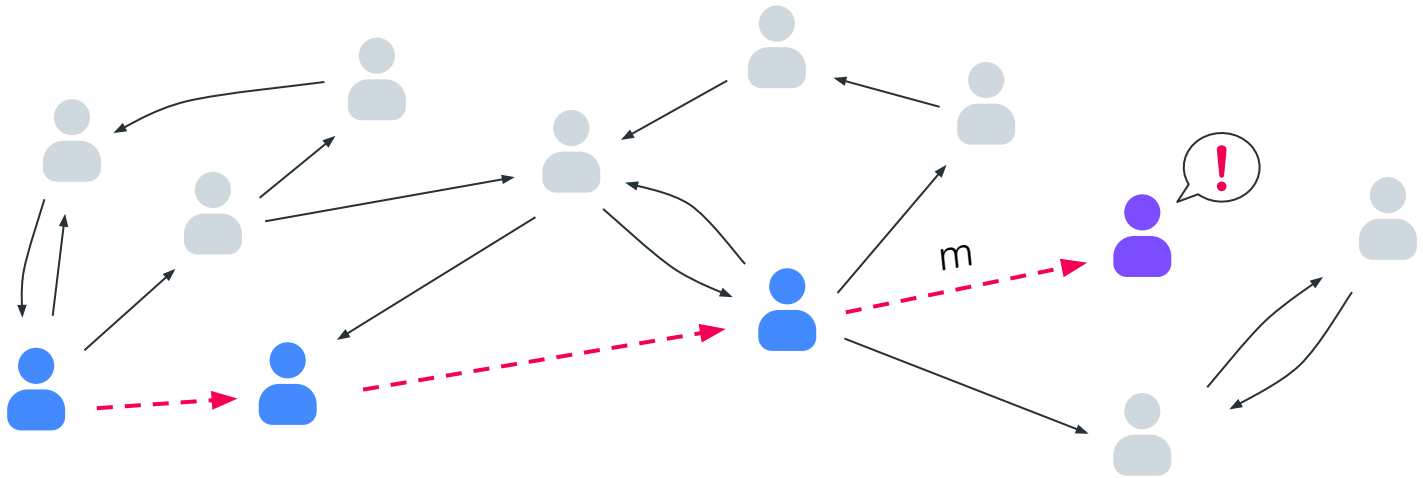
Goal: Reveal limited information after report



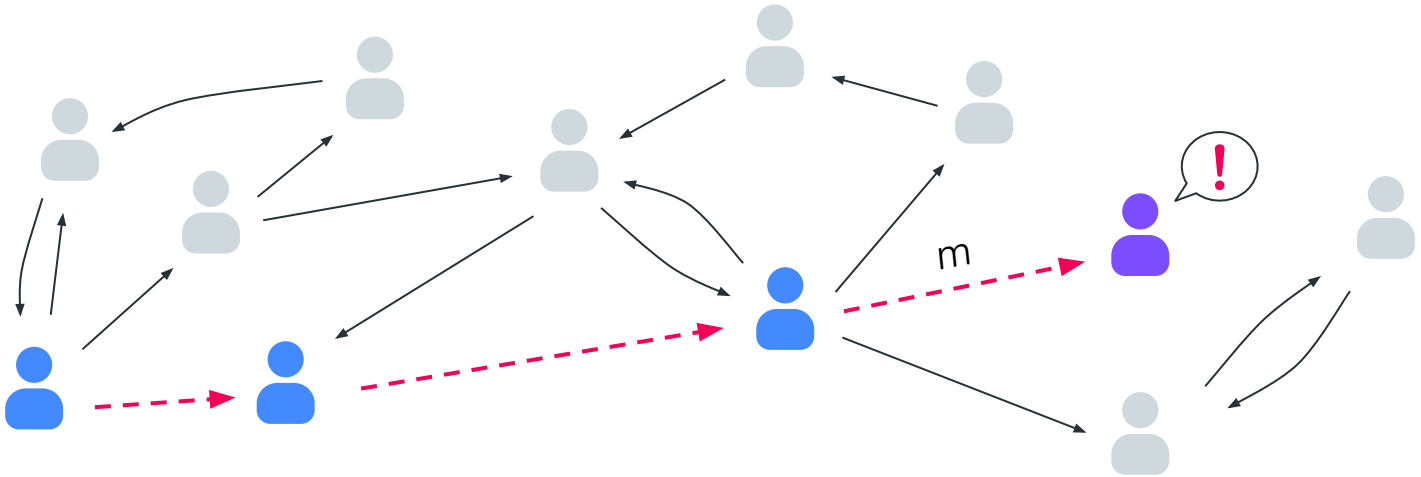
Goal: Reveal limited information after report

After report

Platform view: message content and forward links of traceback target (e.g. path, tree)



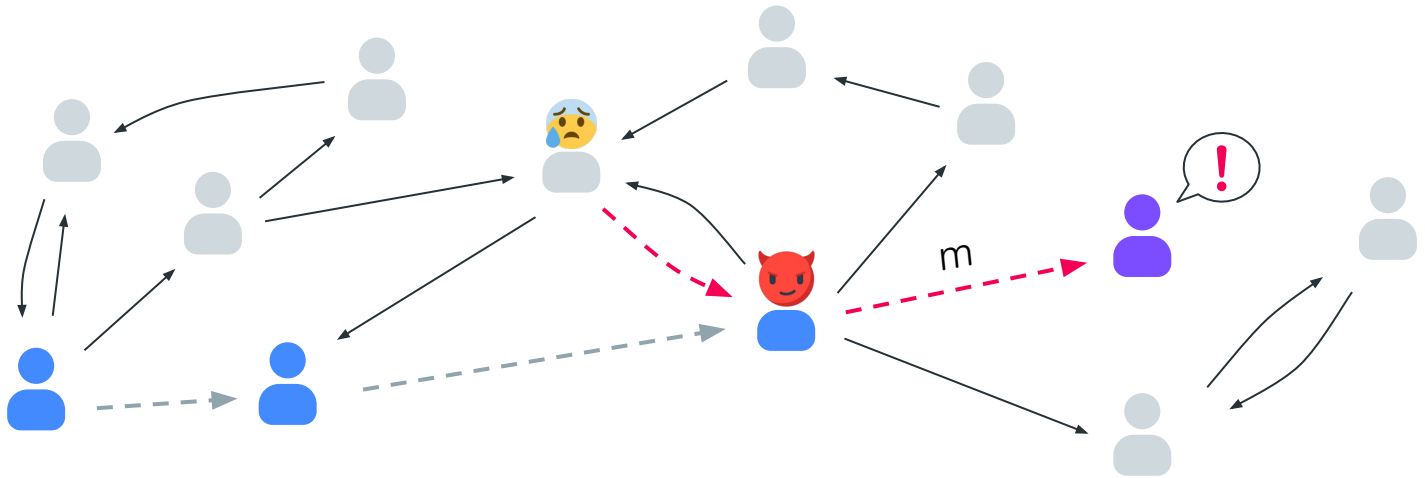
Goal: Report consists of accurate information



Goal: Report consists of accurate information

Trace accountability

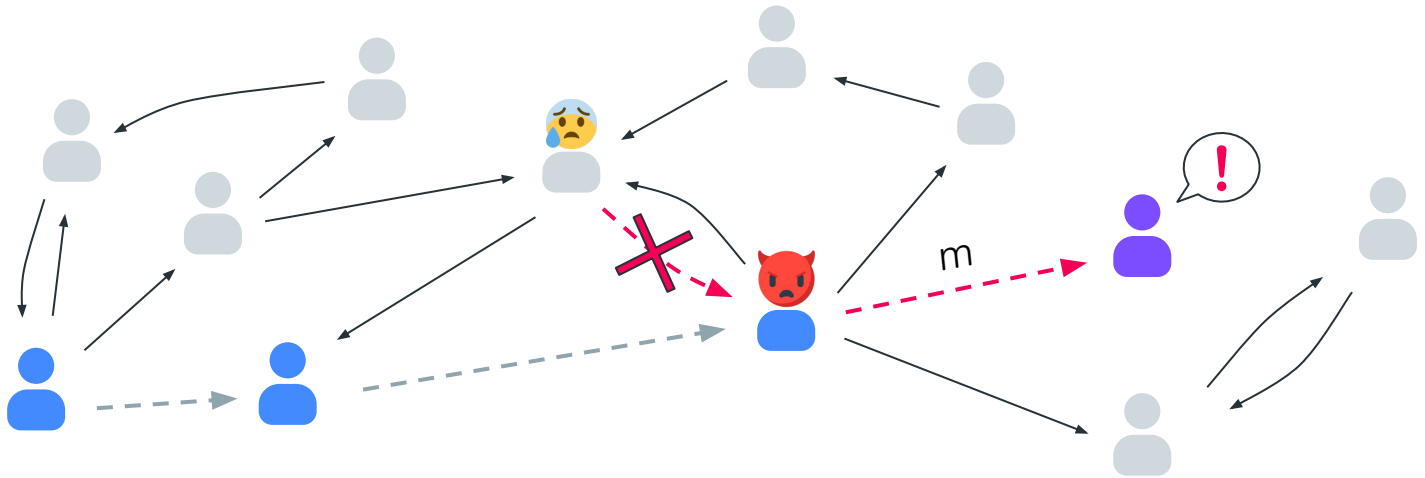
An honest user cannot be framed for an action they didn't perform



Goal: Report consists of accurate information

Trace accountability

An honest user cannot be framed for an action they didn't perform



Malicious user can partition trace, but will be blamed as source

Path traceback

Idea: Linked list of encrypted pointers

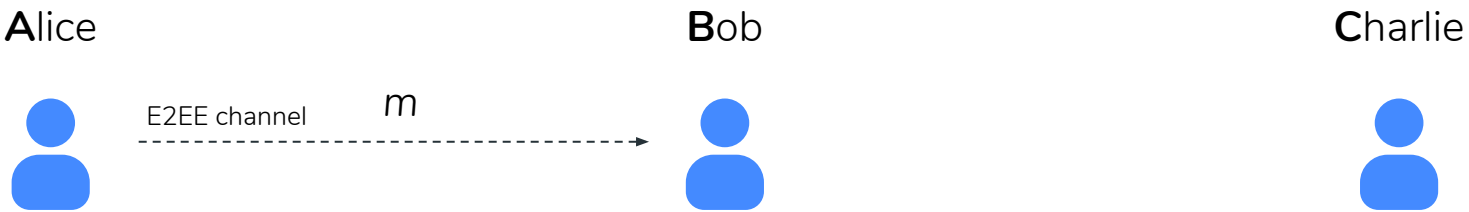
Alice

Bob

Charlie

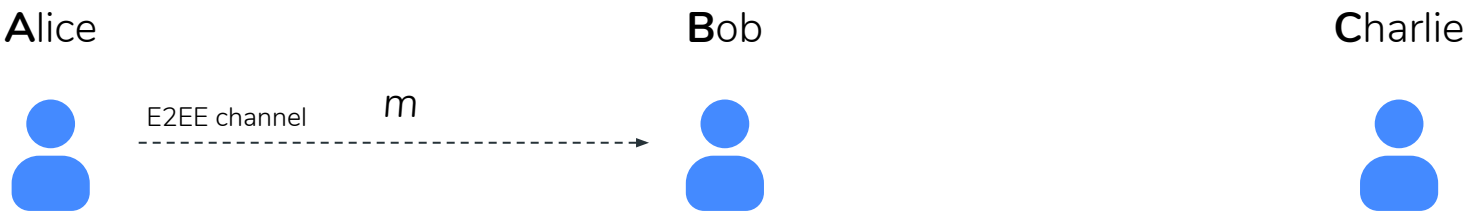
Path traceback

Idea: Linked list of encrypted pointers



Path traceback

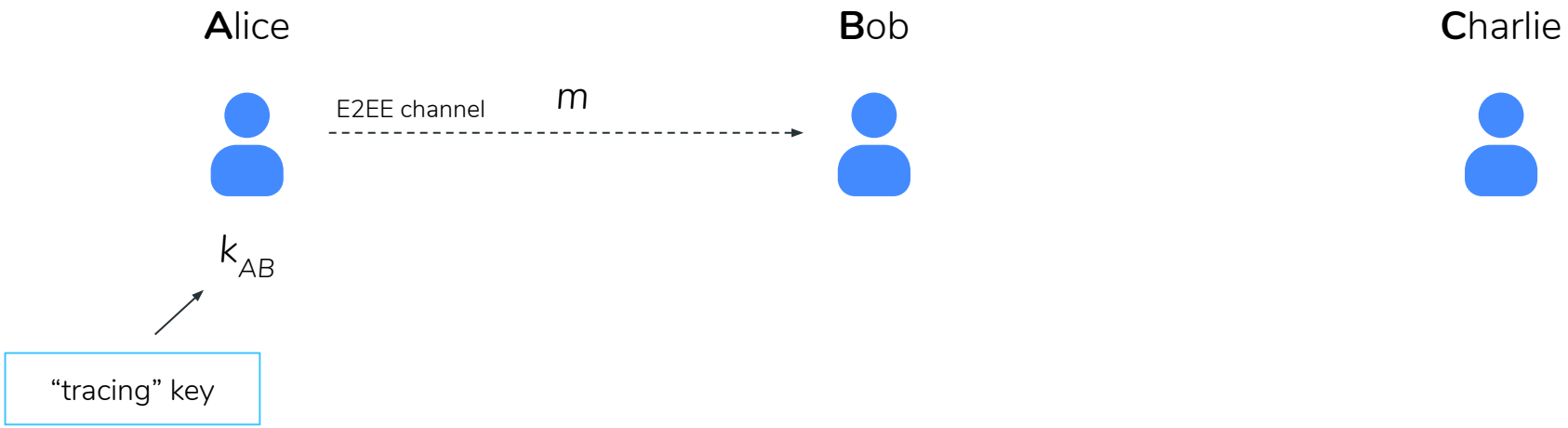
Idea: Linked list of encrypted pointers



- E2EE channel that is decoupled from message tracing

Path traceback

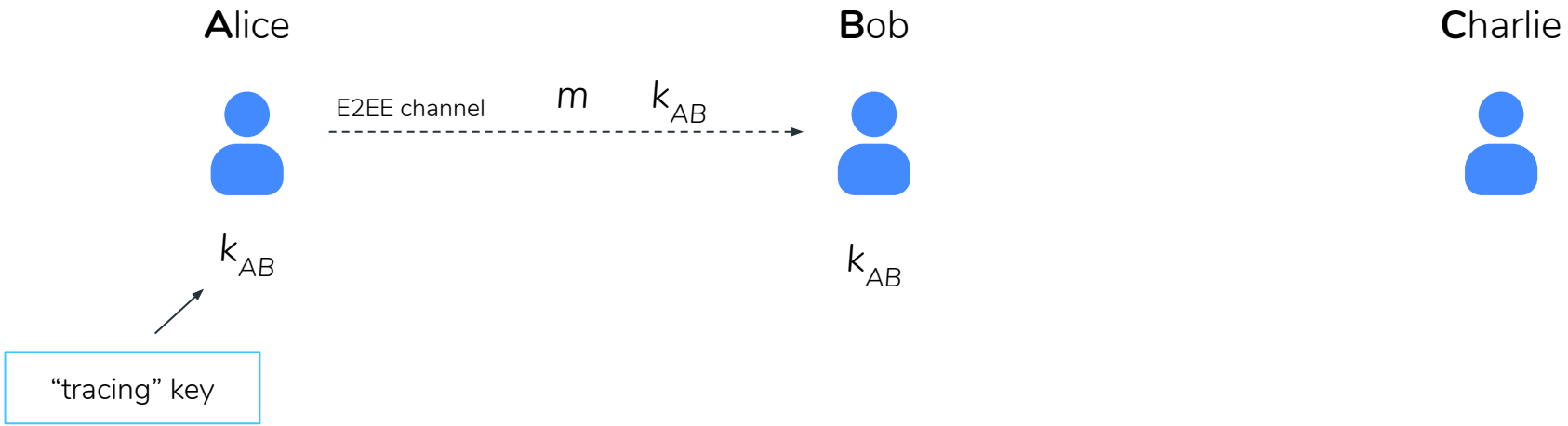
Idea: Linked list of encrypted pointers



- E2EE channel that is decoupled from message tracing
- Unique per-message “tracing” key shared between communication partners

Path traceback

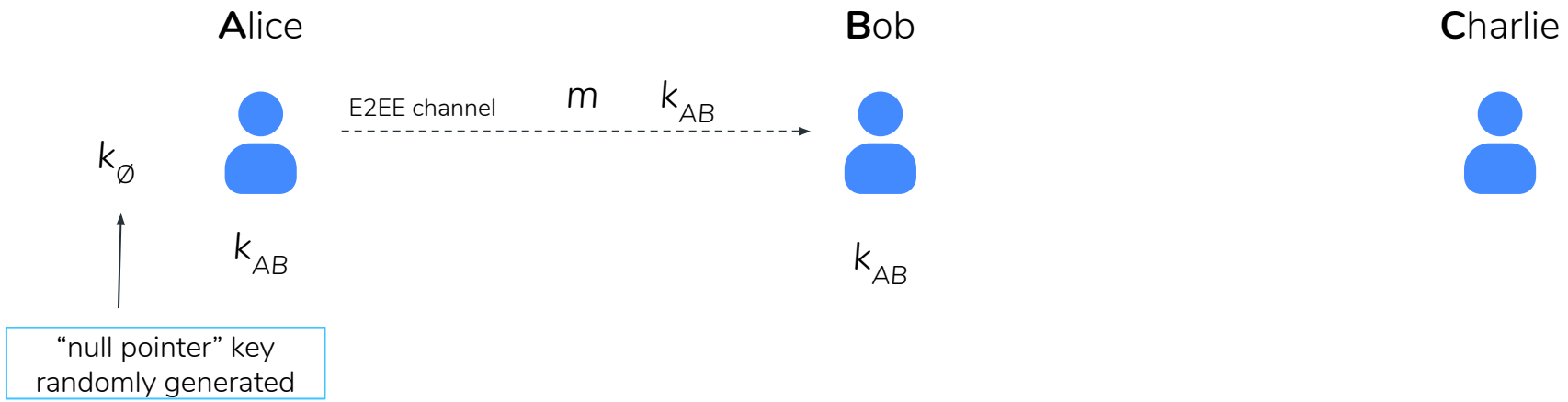
Idea: Linked list of encrypted pointers



- E2EE channel that is decoupled from message tracing
- Unique per-message “tracing” key shared between communication partners

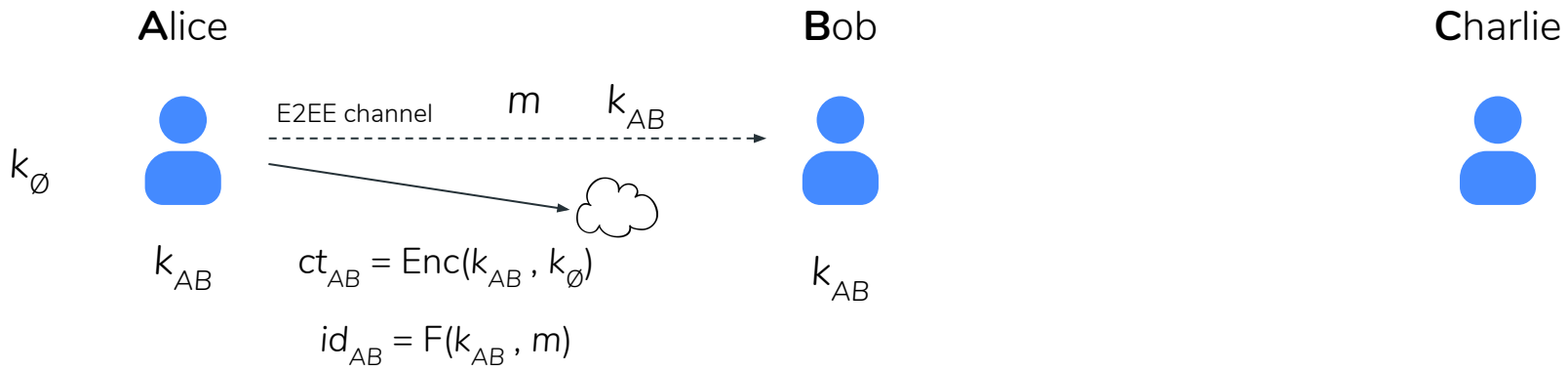
Path traceback

Idea: Linked list of encrypted pointers



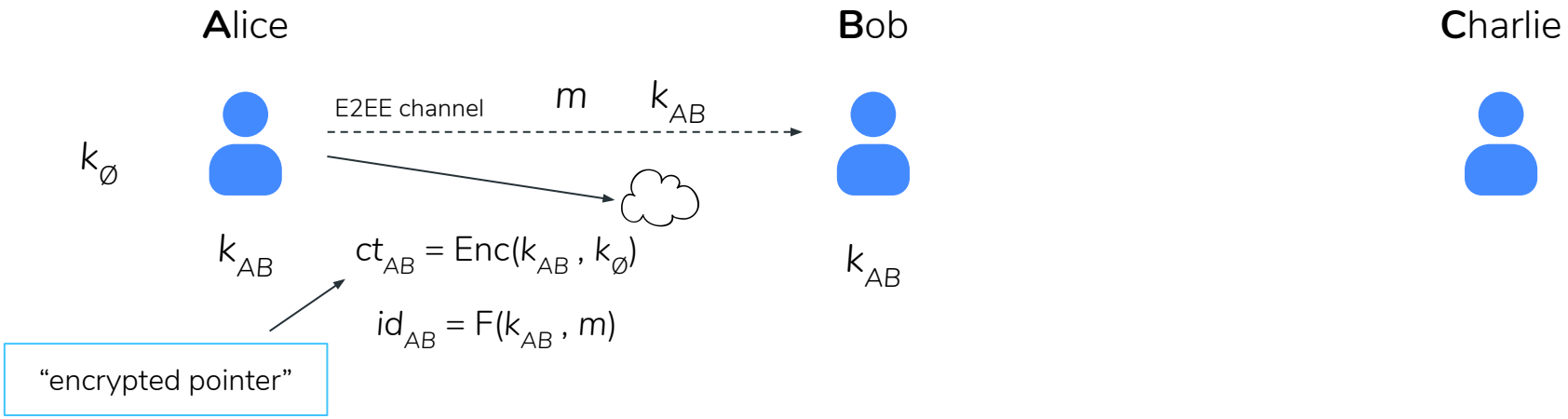
Path traceback

Idea: Linked list of encrypted pointers



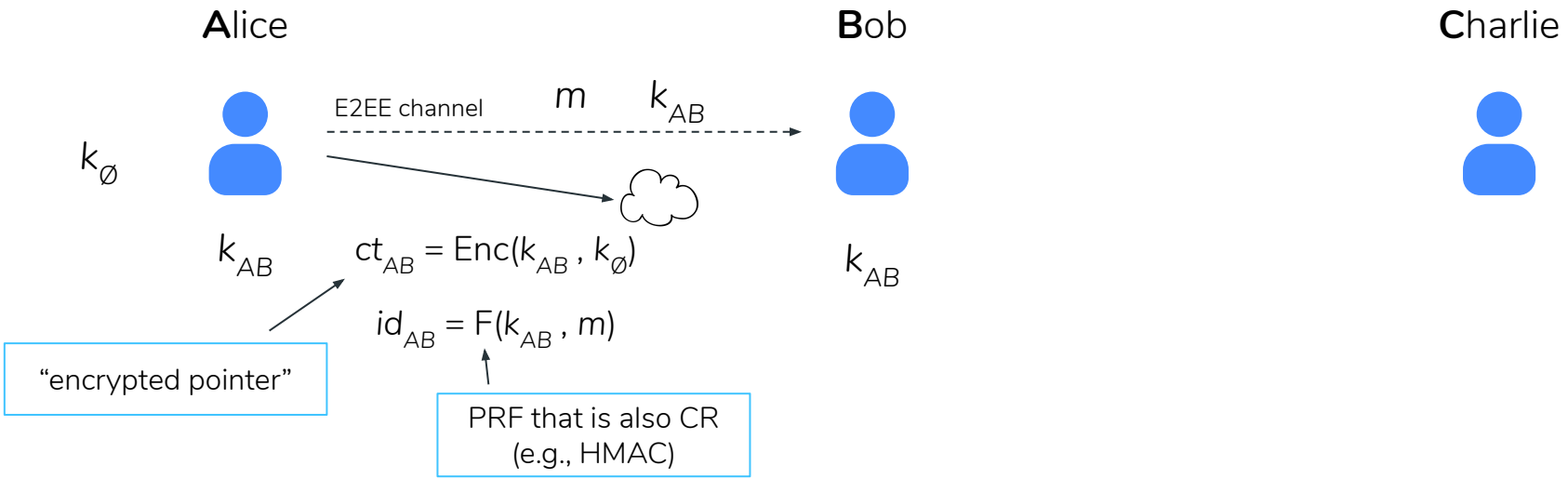
Path traceback

Idea: Linked list of encrypted pointers



Path traceback

Idea: Linked list of encrypted pointers



Path traceback

Idea: Linked list of encrypted pointers

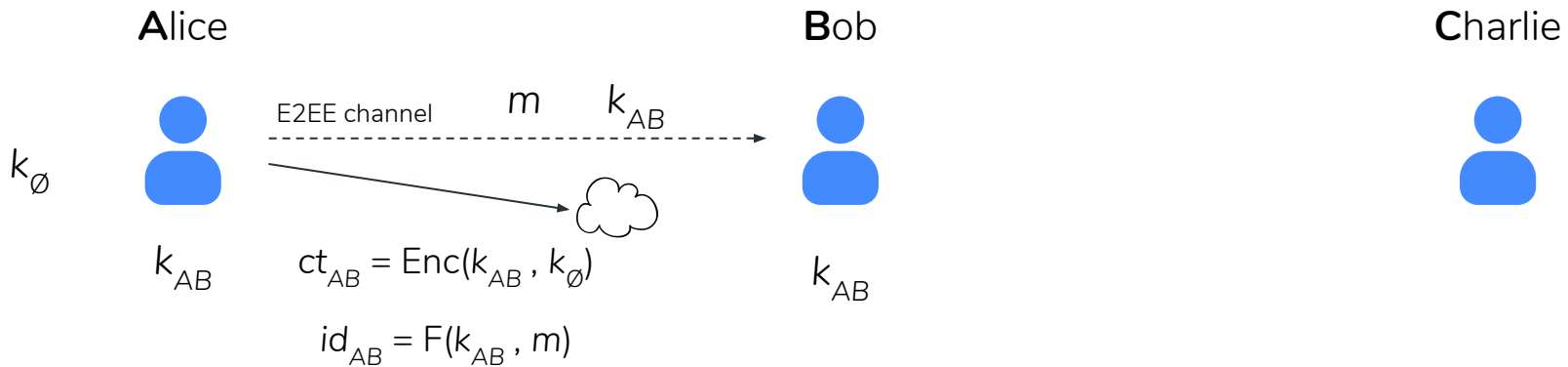


Table stored on platform

id_{AB}	ct_{AB}

Path traceback

Idea: Linked list of encrypted pointers

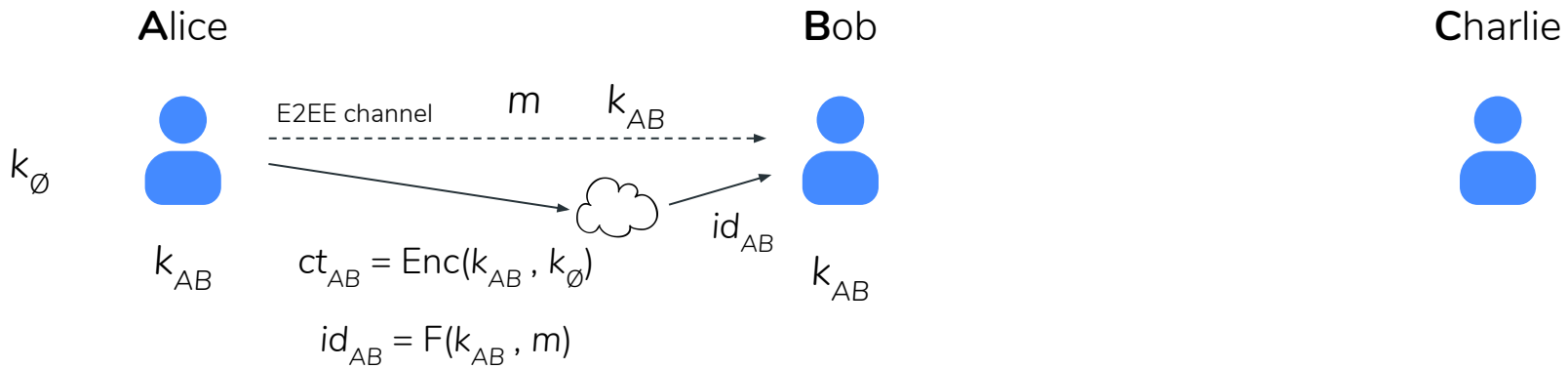


Table stored on platform

id_{AB}	ct_{AB}

Path traceback

Idea: Linked list of encrypted pointers

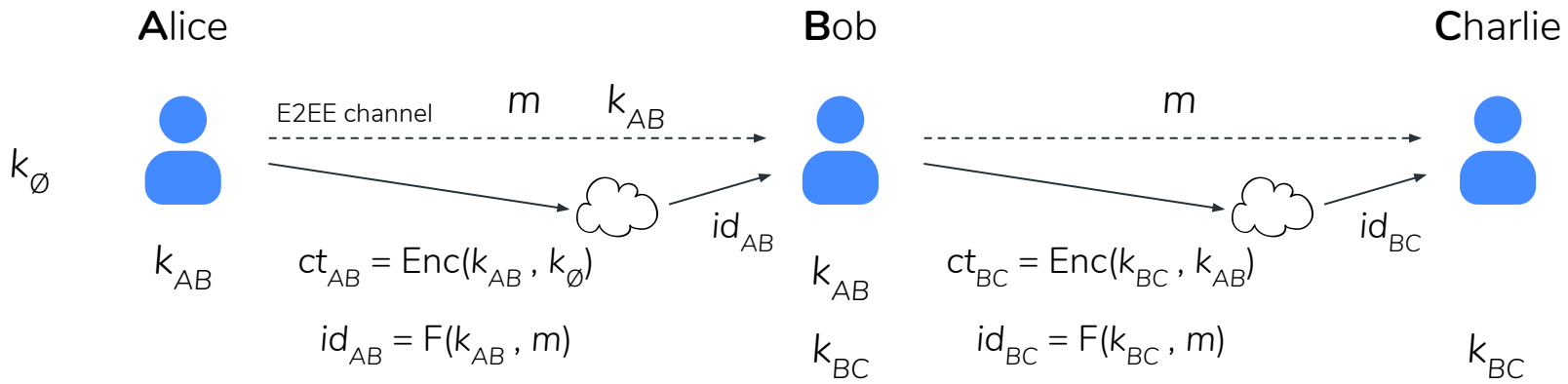


Table stored on platform

id_{AB}	ct_{AB}
id_{BC}	ct_{BC}

Path traceback

Idea: Linked list of encrypted pointers

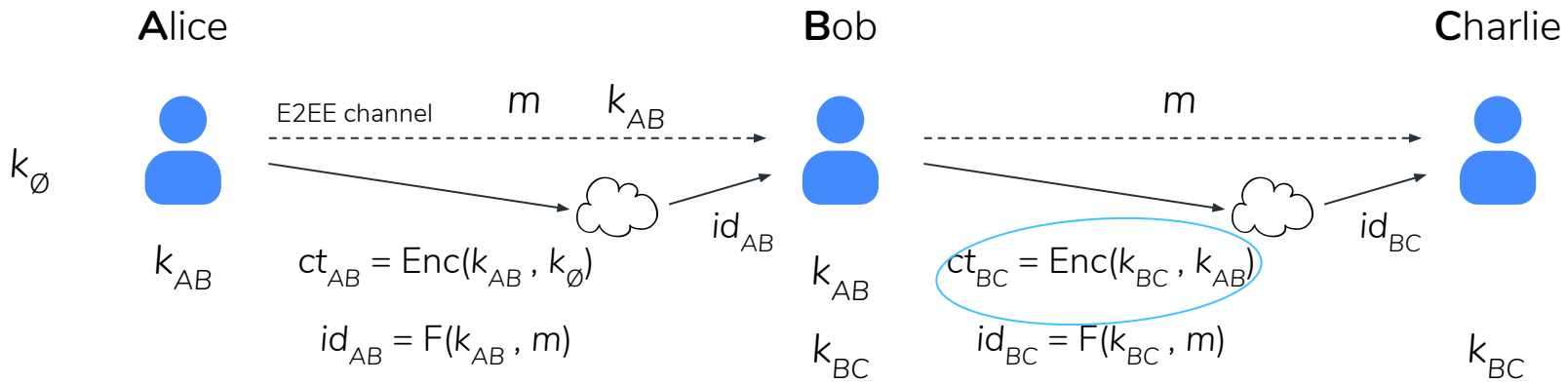


Table stored on platform

id_{AB}	ct_{AB}
id_{BC}	ct_{BC}

Path traceback

Idea: Linked list of encrypted pointers

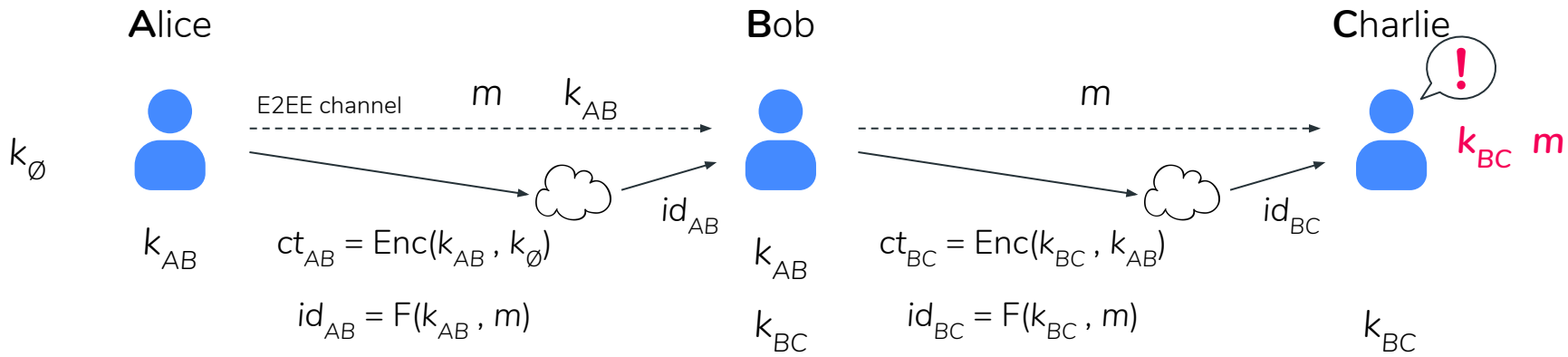


Table stored on platform

id_{AB}	ct_{AB}
id_{BC}	ct_{BC}

Path traceback

Idea: Linked list of encrypted pointers

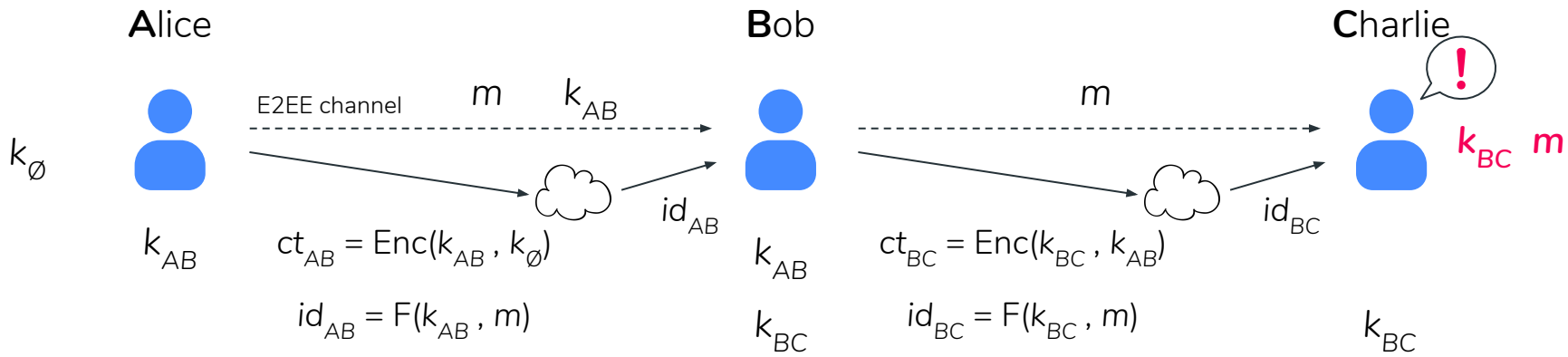


Table stored on platform

	id_{AB}	ct_{AB}
$F(k_{BC}, m)$ →	id_{BC}	ct_{BC}

Path traceback

Idea: Linked list of encrypted pointers

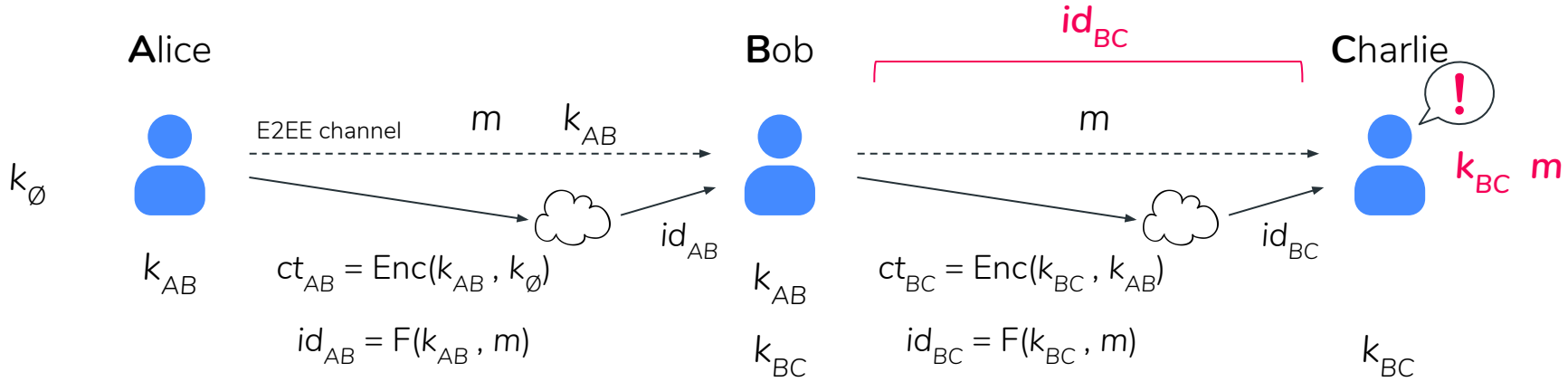


Table stored on platform

	id_{AB}	ct_{AB}
$F(k_{BC}, m)$ →	id_{BC}	ct_{BC}

Path traceback

Idea: Linked list of encrypted pointers

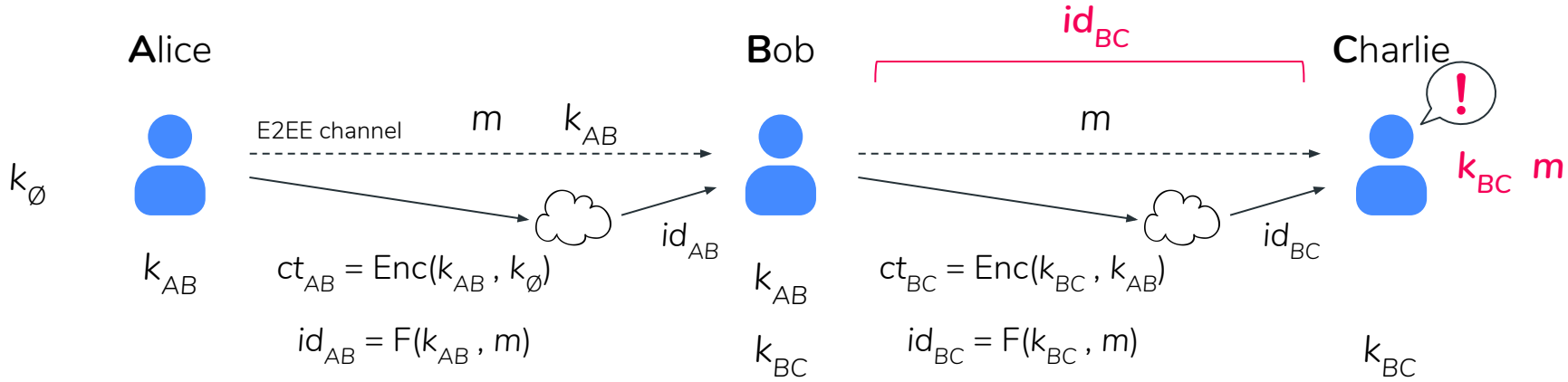


Table stored on platform

id_{AB}	ct_{AB}
id_{BC}	ct_{BC}

$F(k_{BC}, m) \rightarrow$

Decrypt and dereference ct_{BC}

$k_{AB} = Dec(k_{BC}, ct_{BC})$

Path traceback

Idea: Linked list of encrypted pointers

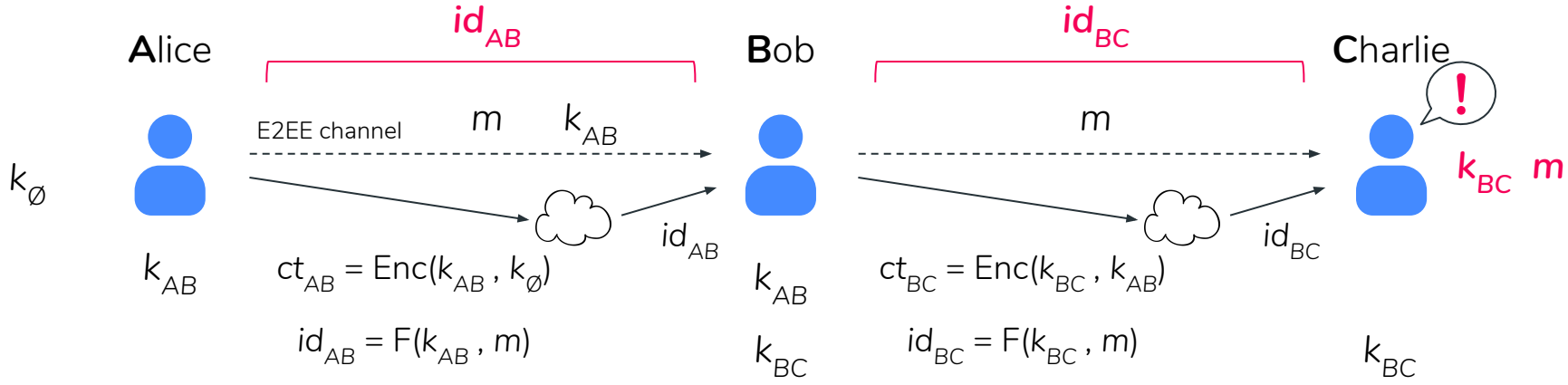


Table stored on platform

$F(k_{AB}, m)$ →	id_{AB}	ct_{AB}
$F(k_{BC}, m)$ →	id_{BC}	ct_{BC}

Decrypt and dereference ct_{BC}

$k_{AB} = Dec(k_{BC}, ct_{BC})$

Path traceback

Idea: Linked list of encrypted pointers

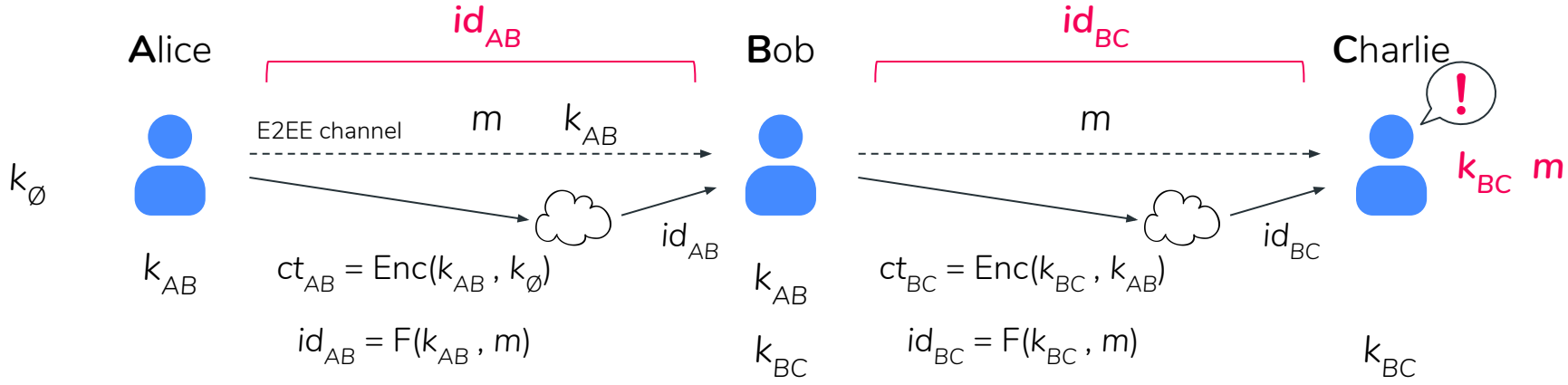


Table stored on platform

	id_{AB}	ct_{AB}
$F(k_{AB}, m)$ →		
$F(k_{BC}, m)$ →	id_{BC}	ct_{BC}

Decrypt and dereference ct_{BC}

$k_{AB} = Dec(k_{BC}, ct_{BC})$

Decrypt and dereference ct_{AB}

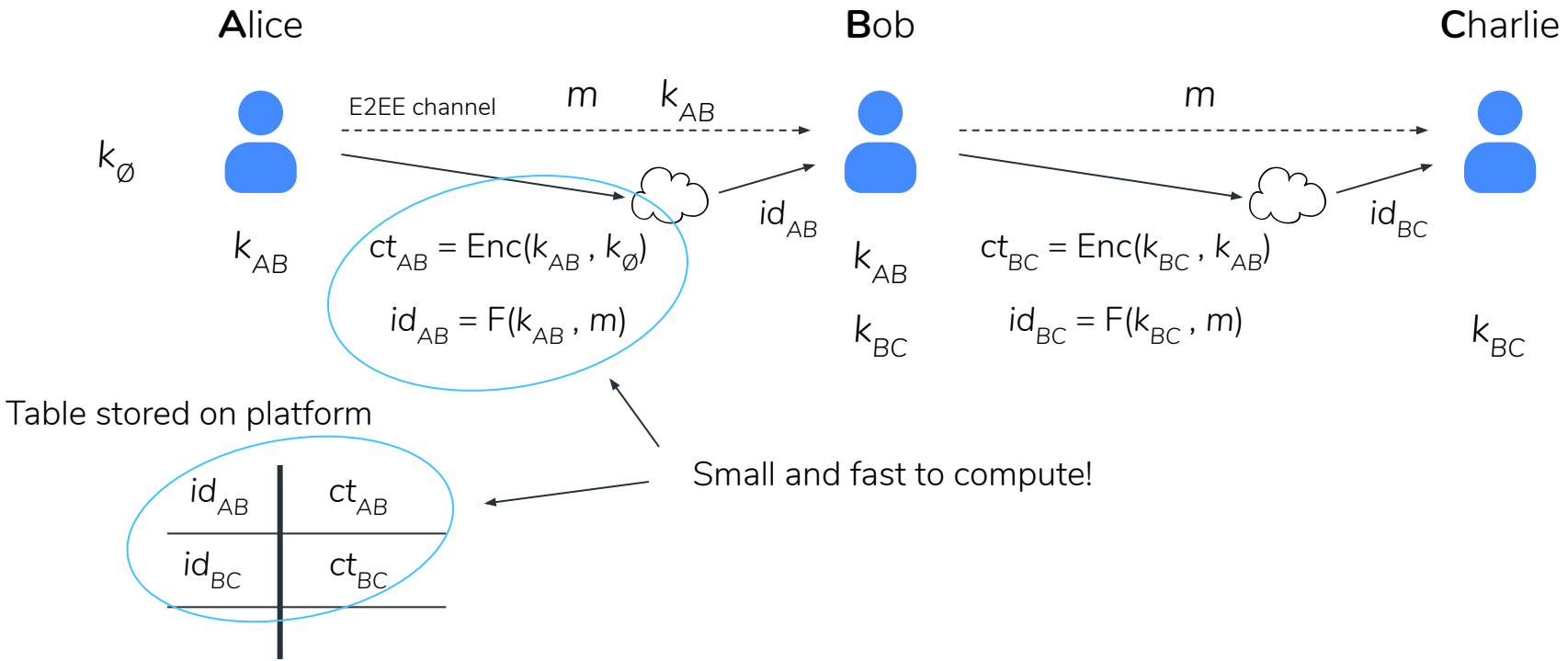
$k_{\emptyset} = Dec(k_{AB}, ct_{AB})$

$F(k_{\emptyset}, m)$ not in table

⇒ beginning of forward chain!

Path traceback

Idea: Linked list of encrypted pointers



Path traceback

Idea: Linked list of encrypted pointers

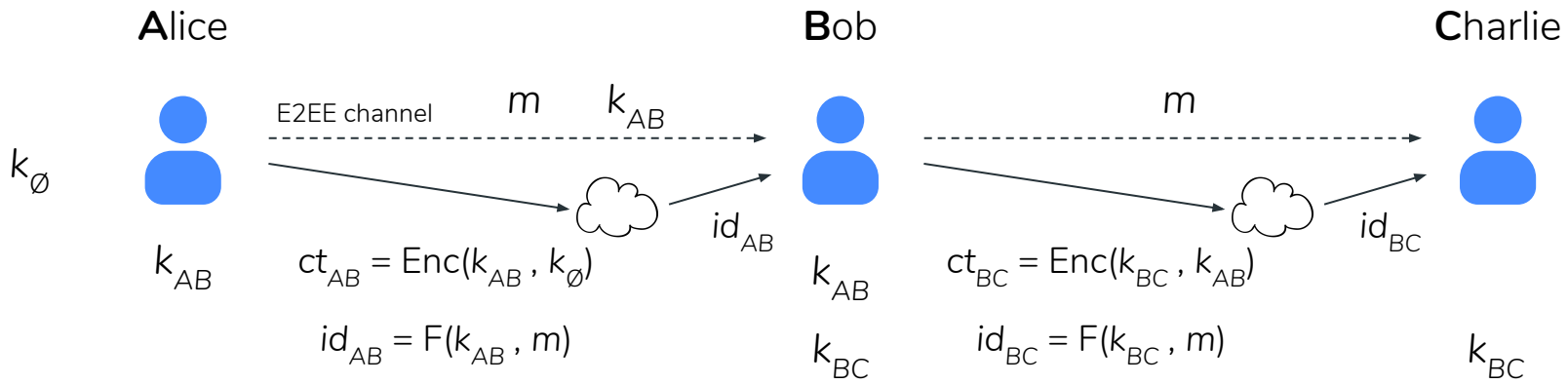
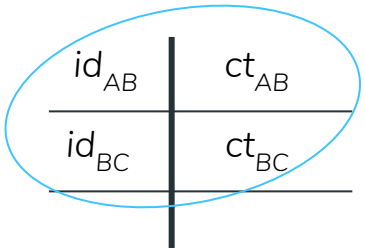


Table stored on platform



Before report

Platform view: Ciphertexts and PRF outputs without keys

User view: Keys without ciphertext

Path traceback

Idea: Linked list of encrypted pointers

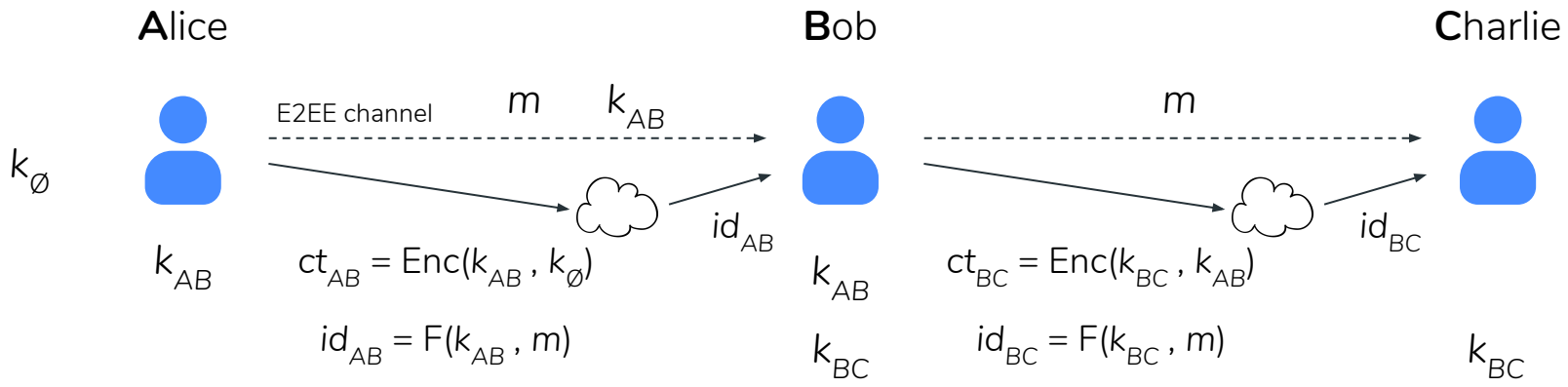
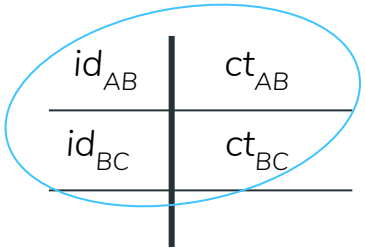


Table stored on platform



Before report
Platform view: Ciphertexts and PRF outputs without keys
User view: Keys without ciphertext

After report
Platform view: Learns keys only for rows of trace

Path traceback

Idea: Linked list of encrypted pointers

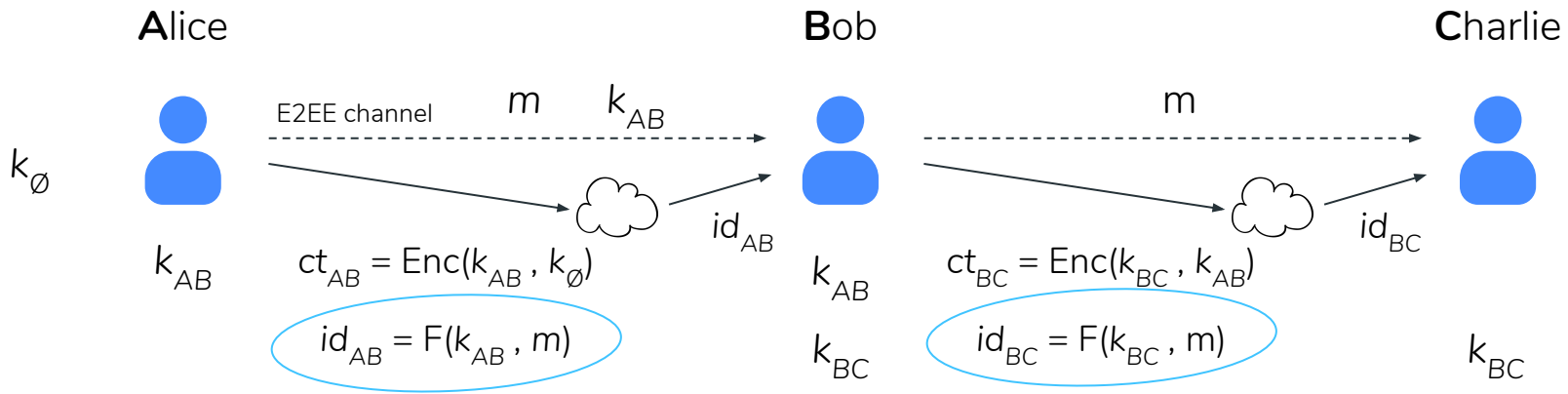


Table stored on platform

id_{AB}	ct_{AB}
id_{BC}	ct_{BC}

Trace accountability
Pointer “dereferences” are bound to a message

Path traceback

Idea: Linked list of encrypted pointers

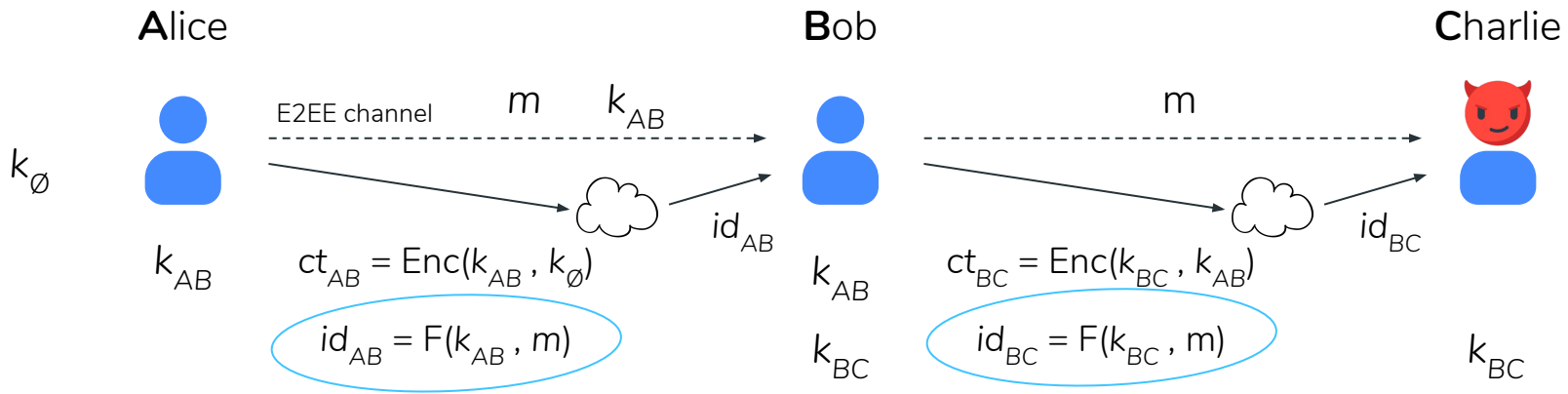


Table stored on platform

id_{AB}	ct_{AB}
id_{BC}	ct_{BC}

Trace accountability
Pointer “dereferences” are bound to a message

Path traceback

Idea: Linked list of encrypted pointers

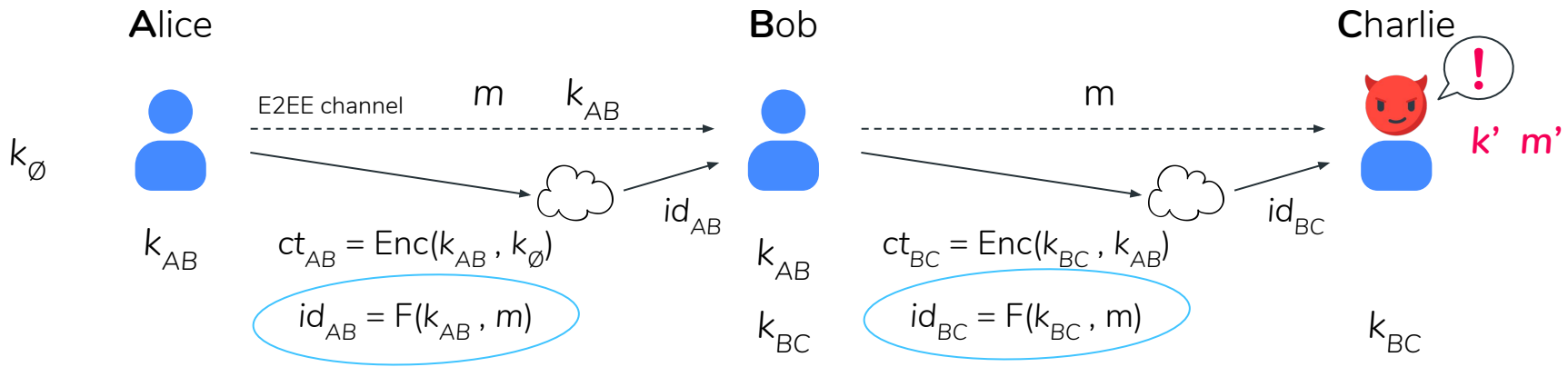


Table stored on platform

id_{AB}	ct_{AB}
id_{BC}	ct_{BC}

Trace accountability
Pointer “dereferences” are bound to a message

Path traceback

Idea: Linked list of encrypted pointers

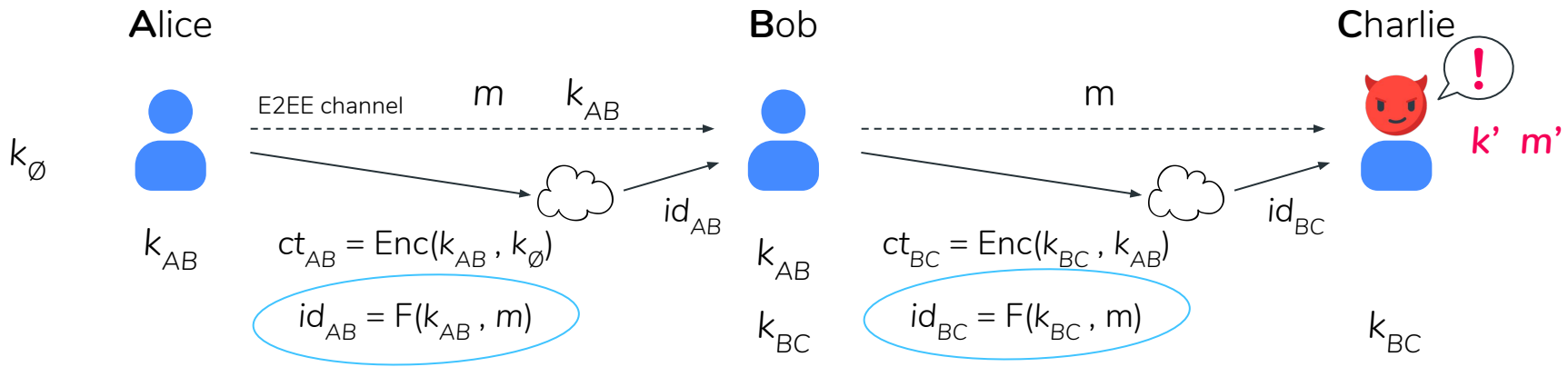


Table stored on platform

id_{AB}	ct_{AB}
id_{BC}	ct_{BC}

Trace accountability
Pointer "dereferences" are bound to a message

To break accountability, $F(k', m')$ must collide with id_{BC}

Path traceback

Idea: Linked list of encrypted pointers

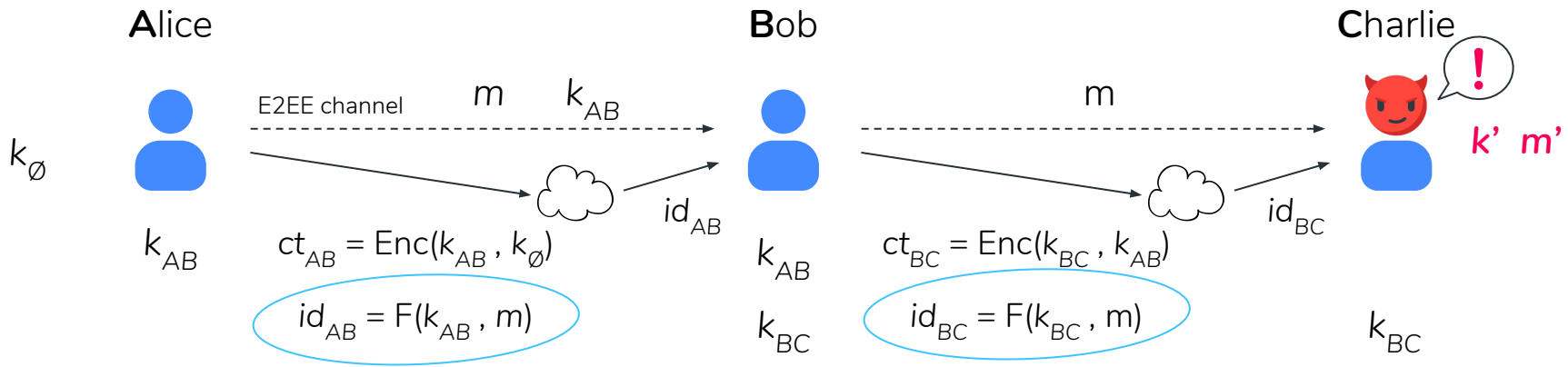


Table stored on platform

id_{AB}	ct_{AB}
id_{BC}	ct_{BC}

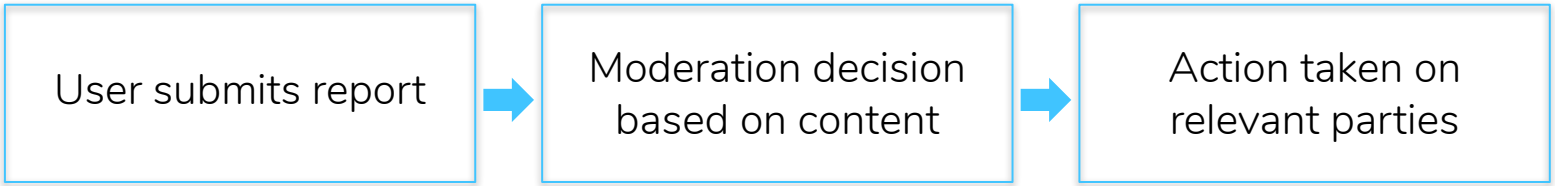
Trace accountability
Pointer "dereferences" are bound to a message

To break accountability, $F(k', m')$ must collide with id_{BC}

See paper for security proofs!

Path traceback

Idea: Linked list of encrypted pointers



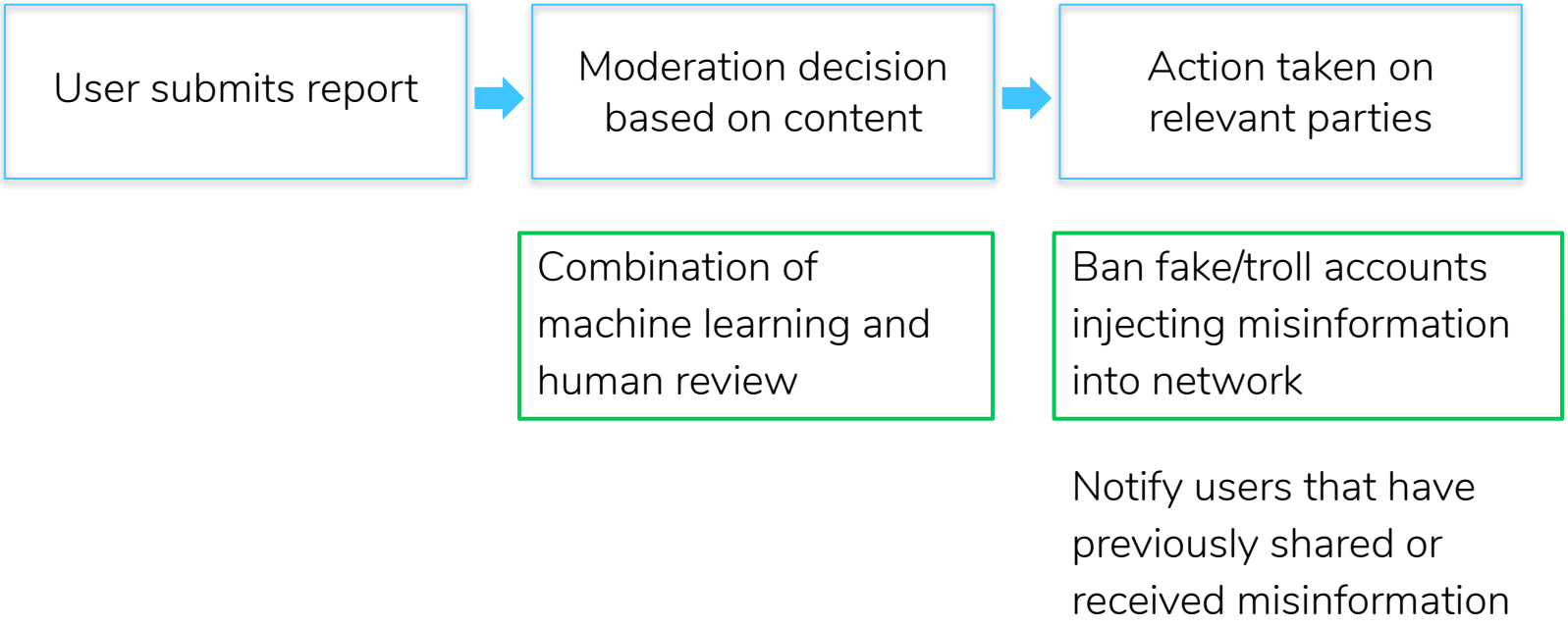
Combination of machine learning and human review

Ban fake/troll accounts injecting misinformation into network

Notify users that have previously shared or received misinformation

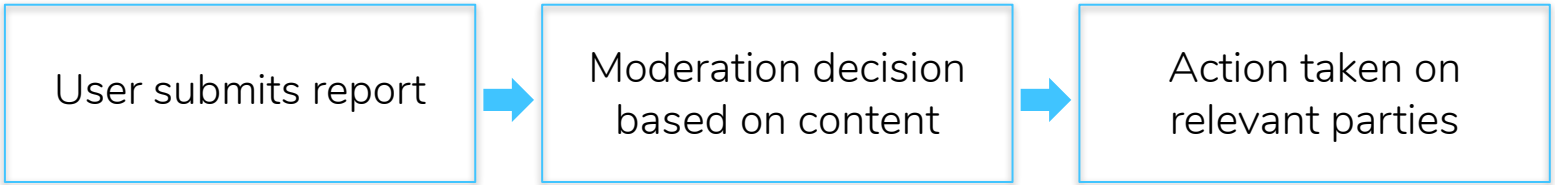
Path traceback

Idea: Linked list of encrypted pointers



Path traceback

Idea: Linked list of encrypted pointers



Combination of machine learning and human review

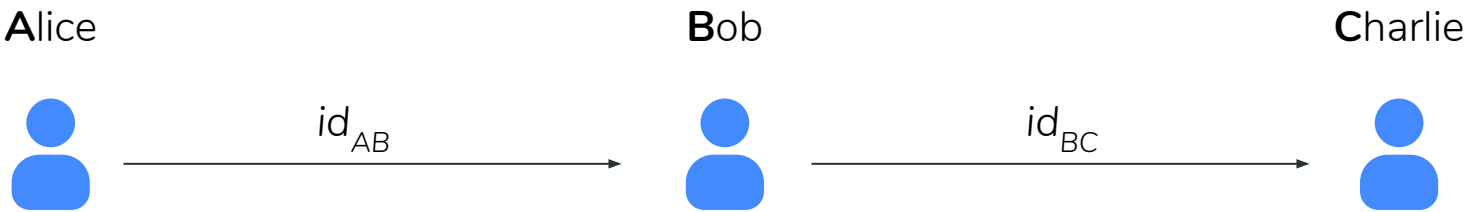
Ban fake/troll accounts injecting misinformation into network

Need something more than path traceback!

Notify users that have previously shared or received misinformation

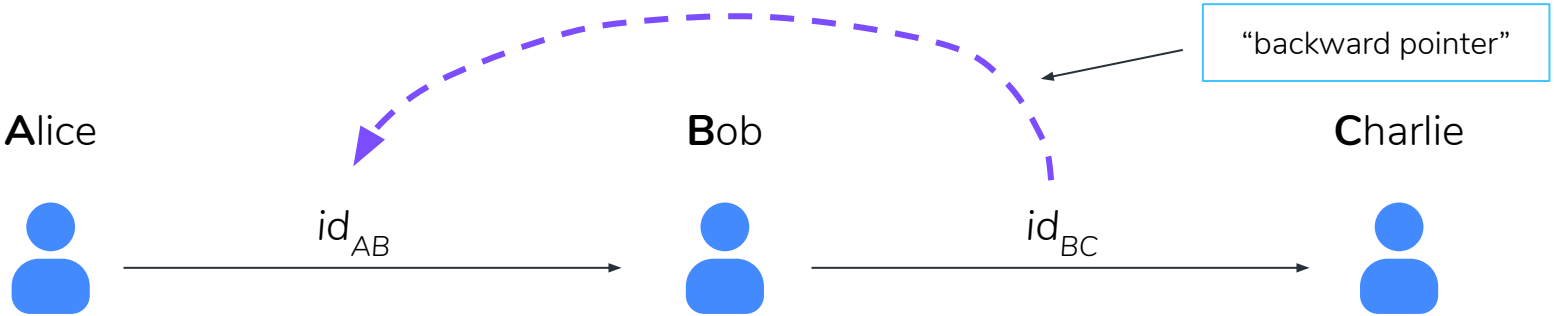
Extension: Tree traceback

Idea: “Doubly” linked list of encrypted pointers



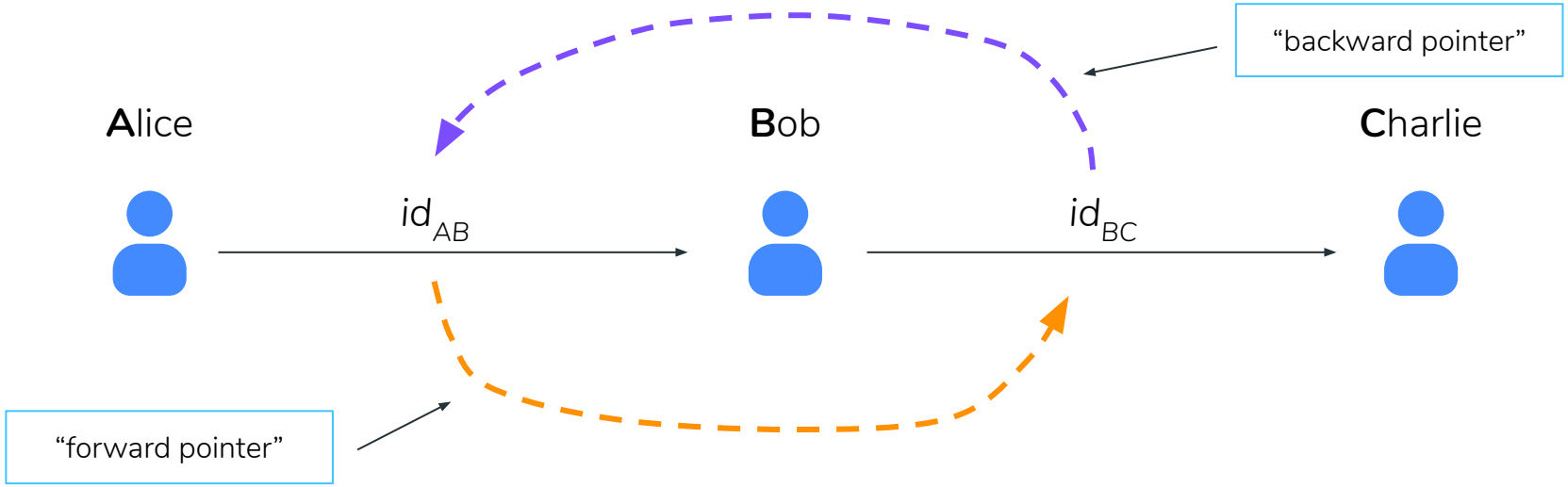
Extension: Tree traceback

Idea: “Doubly” linked list of encrypted pointers



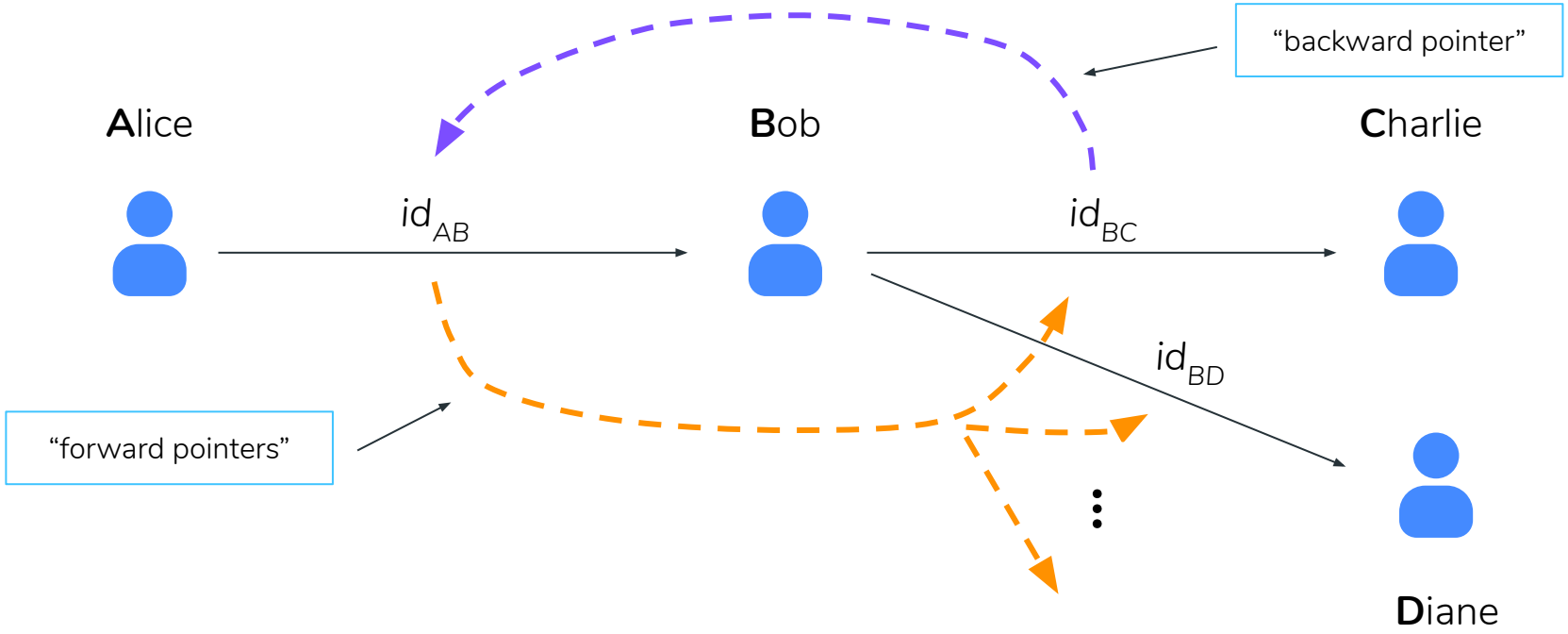
Extension: Tree traceback

Idea: “Doubly” linked list of encrypted pointers



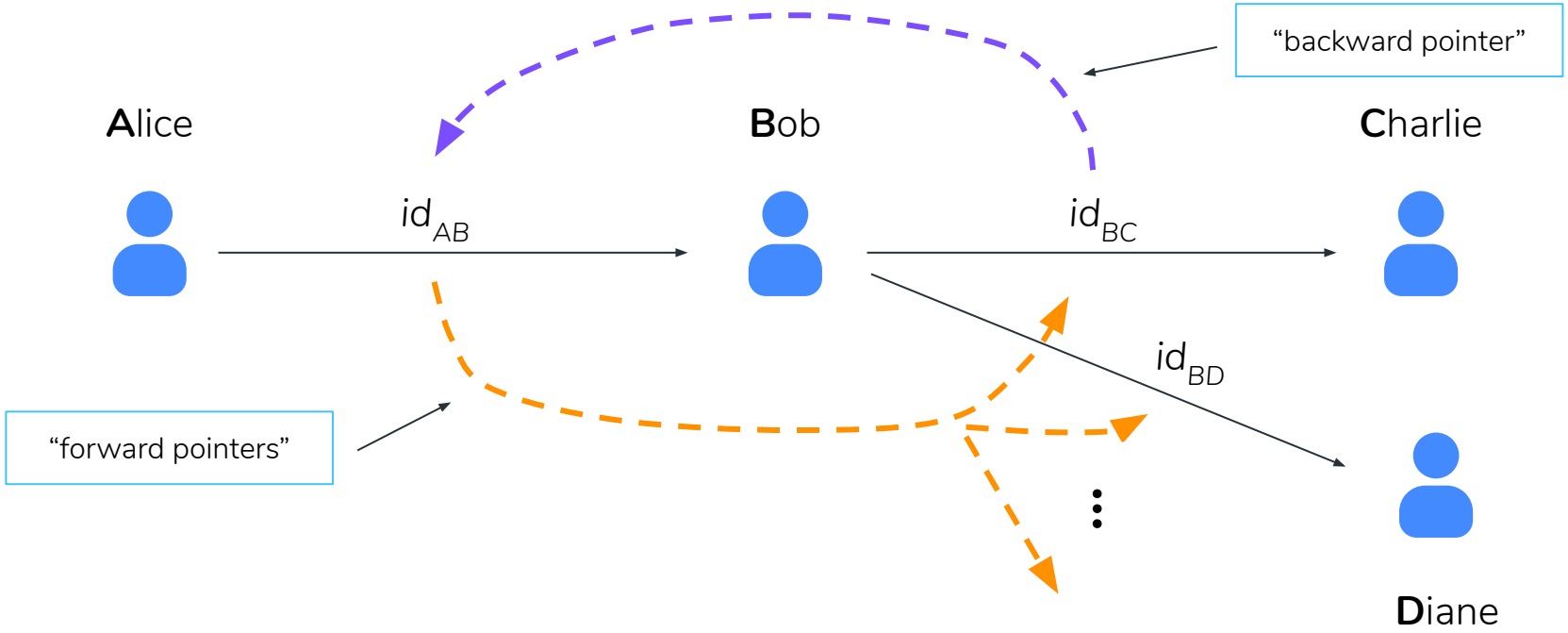
Extension: Tree traceback

Idea: “Doubly” linked list of encrypted pointers



Extension: Tree traceback

Idea: “Doubly” linked list of encrypted pointers



See paper for full details of construction!

(uses PRG and secret sharing)

Performance evaluation

- Path and Tree traceback implemented in < 500 lines of Rust
- Server table stored in in-memory Redis database

https://github.com/nirvantyagi/tracing

Performance evaluation

- Path and Tree traceback implemented in < 500 lines of Rust
- Server table stored in in-memory Redis database

- Fast (uses only efficient symmetric cryptography)
    - Client side: < 50 µs to generate and verify tracing tags
    - Server side: Traceback takes < 100 µs / message in trace

https://github.com/nirvantyagi/tracing

Performance evaluation

- Path and Tree traceback implemented in < 500 lines of Rust
- Server table stored in in-memory Redis database

- Fast (uses only efficient symmetric cryptography)
    - Client side: < 50 µs to generate and verify tracing tags
    - Server side: Traceback takes < 100 µs / message in trace

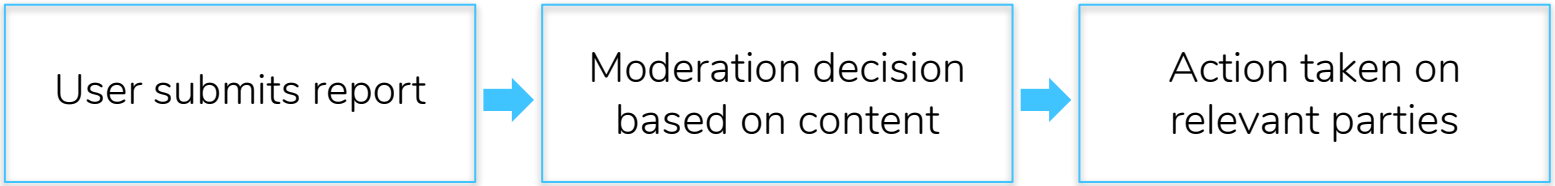
- Platform storage
    - Stores < 100B / message
    - 1 billion messages / day ⇒ ~ 2TB / month
        - Reasonable to store most recent time period sliding window

https://github.com/nirvantyagi/tracing

Deployment considerations

Can tracing be abused to silence socially valuable content?

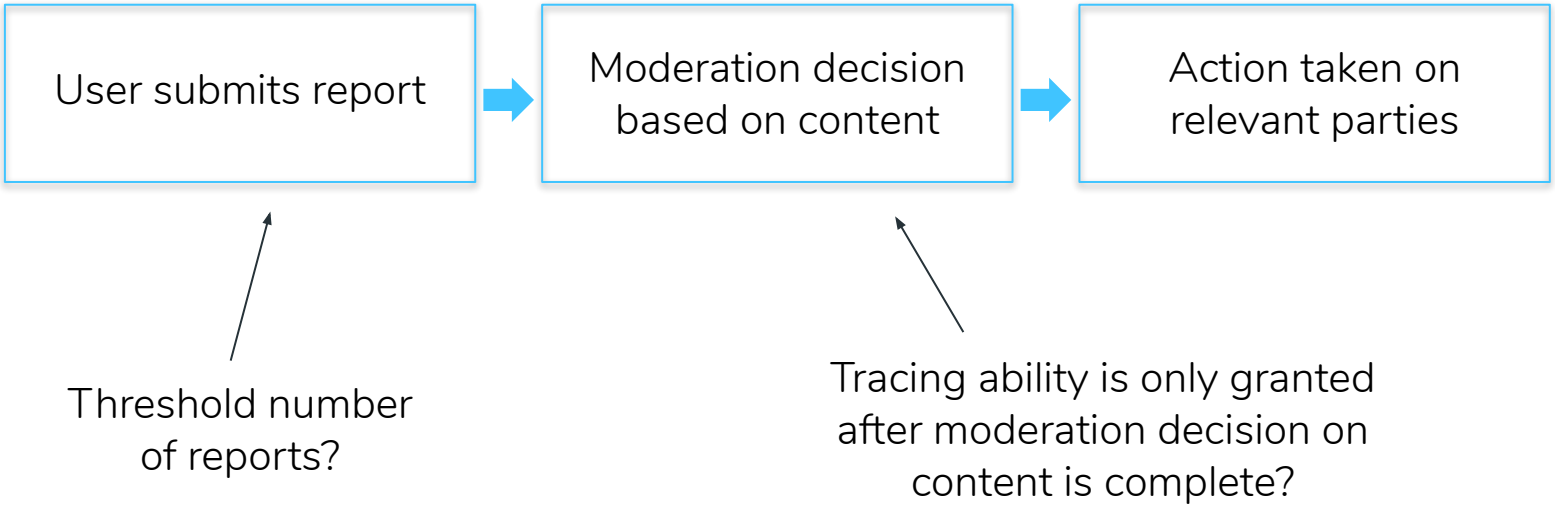
Future work: Policy and implementation to limit abuse of tracing



Deployment considerations

Can tracing be abused to silence socially valuable content?

Future work: Policy and implementation to limit abuse of tracing



Conclusion

- **Message tracing**: new cryptographic functionality for user-driven reporting of forwards in E2EE messaging
    - Path traceback: chain of messages from source to reporter
    - Tree traceback: entire forwarding tree of messages originating from source
- Formal confidentiality and accountability security notions for tracing
- Implementation and evaluation of practicality

https://github.com/nirvantyagi/tracing