Traceback for End-to-End Encrypted Messaging

Nirvan Tyagi  Ian Miers  Tom Ristenpart
Setting: End-to-end encrypted (E2EE) messaging

Alice -> Platform -> Bob

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

Alice

Hello

Platform

> 2 billion users

Bob
Problem: Viral forwarding of misinformation in E2EE messaging
Problem: Viral forwarding of misinformation in E2EE messaging

The New York Times
Disinformation Spreads on WhatsApp Ahead of Brazilian Election
By Mike Isaac and Kevin Roose
Oct. 19, 2018

Wired
How WhatsApp Fuels Fake News and Violence in India
TIMOTHY MCLAUGHLIN
12.12.2018 07:00 AM

The Washington Post
How WhatsApp influenced Nigeria’s recent election — and what it taught us about ‘fake news.’
By Jamie Hitchen, Jonathan Fisher, Nic Cheeseman and Idayat Hassan
Feb. 15, 2019 at 6:00 a.m. EST
Content moderation for user-driven reports

User submits report → Moderation decision based on content → Action taken on relevant parties
Content moderation for user-driven reports

User submits report → Moderation decision based on content → Action taken on relevant parties

Combination of machine learning and human review
Content moderation for user-driven reports

User submits report → Moderation decision based on content → Action taken on relevant parties

- 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

User submits report

Moderation decision based on content

Action taken on relevant parties

Combination of machine learning and human review

Ban fake/troll accounts injecting misinformation into network

Notify users that have previously shared or received misinformation

Report must provide enough information to execute the following steps
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

- **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

- **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

User submits report  ➔  Moderation decision based on content  ➔  Action taken on relevant parties
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]

User reports received message to platform
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

User reports received message to platform
This work: Tracing in E2EE messaging

User reports received message to platform

[TMR CCS’19]
This work: Tracing in E2EE messaging

- Two constructions for message tracing
  - Path traceback

User reports received message to platform
This work: Tracing in E2EE messaging

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

User reports received message to platform
Goal: Act like standard E2EE messaging before report
Goal: Act like standard E2EE messaging

Before report

Platform view: encrypted content and metadata (participants, length, and timing)
Goal: Act like standard E2EE messaging

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

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
Path traceback
Idea: Linked list of encrypted pointers

Alice ——— E2EE channel ——— m ——— Bob ——— Charlie
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

Alice

Bob

Charlie

$k_\emptyset$

$k_{AB}$

$k_{AB}$

$E2EE$ channel

$m$

$k_{AB}$

“null pointer” key
randomly generated
Path traceback

Idea: Linked list of encrypted pointers

\[ ct_{AB} = \text{Enc}(k_{AB}, k_\varnothing) \]
\[ id_{AB} = F(k_{AB}, m) \]
Path traceback

Idea: Linked list of encrypted pointers

k_Ø

k_{AB}

c_{AB} = Enc(k_{AB}, k_Ø)

id_{AB} = F(k_{AB}, m)

"encrypted pointer"
Path traceback
Idea: Linked list of encrypted pointers

$k_\emptyset, k_{AB}$

Alice: $E2EE$ channel, $m$, $k_{AB}$

Bob: $ct_{AB} = Enc(k_{AB}, k_\emptyset)$

Charlie: $k_{AB}$

PRF that is also CR (e.g., HMAC)

"encrypted pointer"
Path traceback

**Idea: Linked list of encrypted pointers**

$k_{0}$

$k_{AB}$

$m$

$k_{AB}$

$ct_{AB} = Enc(k_{AB}, k_{0})$

$id_{AB} = F(k_{AB}, m)$

Table stored on platform

<table>
<thead>
<tr>
<th>$id_{AB}$</th>
<th>$ct_{AB}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Path traceback

Idea: Linked list of encrypted pointers

\[ k_{Ø} \]

Alice

E2EE channel

Bob

\[ m \]

\[ k_{AB} \]

\[ ct_{AB} = \text{Enc}(k_{AB}, k_{Ø}) \]

\[ id_{AB} = F(k_{AB}, m) \]

Charlie

Table stored on platform

<table>
<thead>
<tr>
<th>( id_{AB} )</th>
<th>( ct_{AB} )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Path traceback

Idea: Linked list of encrypted pointers

\[
\begin{align*}
\text{id}_{AB} &= F(k_{AB}, m) \\
\text{ct}_{AB} &= \text{Enc}(k_{AB}, k_\emptyset) \\
\text{id}_{BC} &= F(k_{BC}, m) \\
\text{ct}_{BC} &= \text{Enc}(k_{BC}, k_{AB})
\end{align*}
\]

Table stored on platform

<table>
<thead>
<tr>
<th>id_{AB}</th>
<th>ct_{AB}</th>
</tr>
</thead>
<tbody>
<tr>
<td>id_{BC}</td>
<td>ct_{BC}</td>
</tr>
</tbody>
</table>
Path traceback
Idea: Linked list of encrypted pointers

Table stored on platform

<table>
<thead>
<tr>
<th>$\text{id}_{AB}$</th>
<th>$\text{ct}_{AB}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{id}_{BC}$</td>
<td>$\text{ct}_{BC}$</td>
</tr>
</tbody>
</table>
Path traceback

Idea: Linked list of encrypted pointers

Table stored on platform

<table>
<thead>
<tr>
<th>id_{AB}</th>
<th>ct_{AB}</th>
</tr>
</thead>
<tbody>
<tr>
<td>id_{BC}</td>
<td>ct_{BC}</td>
</tr>
</tbody>
</table>
Path traceback
Idea: Linked list of encrypted pointers

Table stored on platform

<table>
<thead>
<tr>
<th>id\textsubscript{AB}</th>
<th>ct\textsubscript{AB}</th>
</tr>
</thead>
<tbody>
<tr>
<td>id\textsubscript{BC}</td>
<td>ct\textsubscript{BC}</td>
</tr>
</tbody>
</table>
Path traceback

Idea: Linked list of encrypted pointers

Table stored on platform

\[
\begin{array}{c|c}
\text{id}_{AB} & \text{ct}_{AB} \\
\text{id}_{BC} & \text{ct}_{BC}
\end{array}
\]

\[
\text{id}_{AB} = F(\text{k}_{AB}, m)
\]

\[
\text{id}_{BC} = F(\text{k}_{BC}, m)
\]
Path traceback
Idea: Linked list of encrypted pointers

Table stored on platform

<table>
<thead>
<tr>
<th>$id_{AB}$</th>
<th>$ct_{AB}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$id_{BC}$</td>
<td>$ct_{BC}$</td>
</tr>
</tbody>
</table>

Decrypt and dereference $ct_{BC}$

$k_{AB} = \text{Dec}(k_{BC}, ct_{BC})$
Path traceback

Idea: Linked list of encrypted pointers

Alice

$$k_0$$

$$k_{AB}$$

$$ct_{AB} = \text{Enc}(k_{AB}, k_0)$$

$$id_{AB} = F(k_{AB}, m)$$

Bob

$$k_{AB}$$

$$m$$

$$k_{BC}$$

$$ct_{BC} = \text{Enc}(k_{BC}, k_{AB})$$

$$id_{BC} = F(k_{BC}, m)$$

Charlie

$$k_{BC}$$

$$m$$

Table stored on platform

<table>
<thead>
<tr>
<th>$$F(k_{AB}, m)$$</th>
<th>$$id_{AB}$$</th>
<th>$$ct_{AB}$$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$$F(k_{BC}, m)$$</td>
<td>$$id_{BC}$$</td>
<td>$$ct_{BC}$$</td>
</tr>
</tbody>
</table>

Decrypt and dereference $$ct_{BC}$$

$$k_{AB} = \text{Dec}(k_{BC}, ct_{BC})$$
Path traceback

Idea: Linked list of encrypted pointers

Table stored on platform

<table>
<thead>
<tr>
<th>Functionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decrypt and dereference $ct_{BC}$</td>
</tr>
<tr>
<td>$k_{AB} = \text{Dec}(k_{BC}, ct_{BC})$</td>
</tr>
<tr>
<td>$k_\emptyset = \text{Dec}(k_{AB}, ct_{AB})$</td>
</tr>
<tr>
<td>$F(k_\emptyset, m)$ not in table</td>
</tr>
<tr>
<td>$F(k_{BC}, m)$ beginning of forward chain!</td>
</tr>
</tbody>
</table>
Path traceback

Idea: Linked list of encrypted pointers

Alice

\[ k_\varnothing \]

\[ k_{AB} \]

\[ ct_{AB} = \text{Enc}(k_{AB}, k_\varnothing) \]

\[ id_{AB} = F(k_{AB}, m) \]

E2EE channel

Bob

\[ k_{AB} \]

\[ id_{AB} \]

\[ ct_{BC} = \text{Enc}(k_{BC}, k_{AB}) \]

\[ id_{BC} = F(k_{BC}, m) \]

Charlie

\[ k_{BC} \]

Table stored on platform

Small and fast to compute!
Path traceback
Idea: Linked list of encrypted pointers

Table stored on platform

<table>
<thead>
<tr>
<th>id_{AB}</th>
<th>ct_{AB}</th>
</tr>
</thead>
<tbody>
<tr>
<td>id_{BC}</td>
<td>ct_{BC}</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>id_{AB}</th>
<th>ct_{AB}</th>
</tr>
</thead>
<tbody>
<tr>
<td>id_{BC}</td>
<td>ct_{BC}</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>id_{AB}</th>
<th>ct_{AB}</th>
</tr>
</thead>
<tbody>
<tr>
<td>id_{BC}</td>
<td>ct_{BC}</td>
</tr>
</tbody>
</table>
Path traceback

Idea: Linked list of encrypted pointers

Table stored on platform

<table>
<thead>
<tr>
<th>(\text{id}_{AB})</th>
<th>(\text{ct}_{AB})</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\text{id}_{BC})</td>
<td>(\text{ct}_{BC})</td>
</tr>
</tbody>
</table>

Trace accountability

Pointer “dereferences” are bound to a message
Path traceback

Idea: Linked list of encrypted pointers

Table stored on platform

<table>
<thead>
<tr>
<th>id_{AB}</th>
<th>ct_{AB}</th>
<th>id_{BC}</th>
<th>ct_{BC}</th>
</tr>
</thead>
</table>

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

<table>
<thead>
<tr>
<th>id_{AB}</th>
<th>ct_{AB}</th>
</tr>
</thead>
<tbody>
<tr>
<td>id_{BC}</td>
<td>ct_{BC}</td>
</tr>
</tbody>
</table>

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

1. User submits report
2. Moderation decision based on content
3. Action taken on relevant parties

- 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

User submits report

Moderation decision based on content

Action taken on relevant parties

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

User submits report → Moderation decision based on content → Action taken on relevant parties

- Combination of machine learning and human review
- Ban fake/troll accounts injecting misinformation into network
- Notify users that have previously shared or received misinformation

Need something more than path traceback!
Extension: Tree traceback
Idea: “Doubly” linked list of encrypted pointers
Extension: Tree traceback
Idea: “Doubly” linked list of encrypted pointers

Alice \(\rightarrow id_{AB}\) Bob \(\rightarrow id_{BC}\) Charlie

“backward pointer”
Extension: Tree traceback
Idea: “Doubly” linked list of encrypted pointers

A  lice     id_{AB}     B ob     id_{BC}     C harlie

“forward pointer”

“backward pointer”
Extension: Tree traceback

Idea: “Doubly” linked list of encrypted pointers

Alice → Bob → Charlie

“backward pointer”

“forward pointers”

id_{AB} → id_{BC} → id_{BD}
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

User submits report → Moderation decision based on content → Action taken on relevant parties
Deployment considerations

Can tracing be abused to silence socially valuable content?

Future work: Policy and implementation to limit abuse of tracing

- User submits report
- Moderation decision based on content
- Action taken on relevant parties

Threshold number of reports?

Tracing ability is only granted after moderation decision on content is complete?
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