Data provenance in a distributed calculus

Vladimiro Sassone

work in progress with A. Francalanza, J. Rathke, and I. Souilah
Motivation

✧ (Meta)data is almost entirely neglected in the process calculi literature
✧ Track data provenance both for its important applications and as an challenging exercise in modelling (meta)data. We aim at simplicity:
   ✧ data annotations representing provenance
   ✧ structure, interpretation and management of provenance information
   ✧ provenance tracking
✧ Provenance-based security (aspects: trust + data confidentiality and privacy)
   ✧ Example: photography competition
✧ The overall ambition is to underpin practical development, like trust-policy languages and protocols, and provenance-middleware
Model features

两大基本模型的核心特征：

1. 值被注释与它们的起源
2. 起源保持与计算过程同步

专注于一种特定的起源信息：

- 影响一个值的主体，以及它们如何做到这一点。
Provenance model

Annotated data

Annotated value

Value

Actual data

Meta information describing the origin of the value

Provenance

\( \nu : K \)
Provenance model
Structure and interpretation of provenance

"Operations" that were performed on the value. They record the principals that "influenced" the value and how.
Provenance model
Structure and interpretation of provenance

\( \varepsilon \) (empty provenance) denotes value \( v \) originated here

\[ v : \varepsilon ; a!\kappa_1 ; b?\kappa_2 ; b!(\varepsilon;c!\kappa_3,b?\kappa_4) ; \ldots \]
Provenance model
Structure and interpretation of provenance

$\epsilon$ (empty provenance) denotes value $v$ originated here

$v : \epsilon ; a!\kappa_1 ; b?\kappa_2 ; b!(\epsilon; c!\kappa_3, b?!\kappa_4) ; ...$

It was sent by $a$ on a channel with provenance $\kappa_1$
Provenance model
Structure and interpretation of provenance

\[
\nu : \varepsilon ; a!\kappa_1 ; b?\kappa_2 ; b!(\varepsilon ; c!\kappa_3 , b?\kappa_4 ) ; \ldots
\]

\(\varepsilon\) (empty provenance) denotes value \(\nu\) originated here

It was sent by \(a\) on a channel with provenance \(\kappa_1\)

Was then received by \(b\) on a channel with provenance \(\kappa_2\)
Provenance model
Structure and interpretation of provenance

\( \varepsilon \) (empty provenance) denotes value \( v \) originated here

And then sent by \( b \) on a channel that \( b \) received from \( c \)…

\( v : \varepsilon ; a! \kappa_1 ; b? \kappa_2 ; b!(\varepsilon ; c! \kappa_3 , b? \kappa_4 ) ; … \)

It was sent by \( a \) on a channel with provenance \( \kappa_1 \)

Was then received by \( b \) on a channel with provenance \( \kappa_2 \)
Provenance model

Provenance tracking

\[ \begin{align*}
  n : \kappa_n & ! \langle v : \kappa_v \rangle \\
  n : \kappa'_n & ?(x).P \\
  P \{ v : \kappa_v ; a! \kappa_n ; b? \kappa'_n \slash x \}
\end{align*} \]
Provenance model

Provenance tracking

Principals

\[
\begin{array}{c}
n : \kappa_n \vdash \langle \nu : \kappa_v \rangle \\
n : \kappa'_n \vdash ?(x).P \\
\end{array}
\]

\[
P\{v : \kappa_v ; a!\kappa_n ; b?\kappa'_n \} / x
\]
Provenance model

Provenance tracking

Communication channels

\[ n : \kappa_n ! \langle v : \kappa_v \rangle \]

\[ n : \kappa_n' ?(x).P \]

\[ P \{ v : \kappa_v ; a!\kappa_n ; b?\kappa_n' / x \} \]
Channels, like all other values, are annotated with their provenance.

\[ n : \kappa_n ! \langle v : \kappa_v \rangle \]

\[ n : \kappa'_n ?(x).P \]

\[ P\{v : \kappa_v ; a!\kappa_n ; b?\kappa'_n / x\} \]
Provenance model

Annotated value to be sent

Placeholder for value to be received

Value received by \( b \)
Provenance model
Provenance tracking

Old provenance of $\nu$

New provenance of $\nu$

$n : \kappa_n ! \langle \nu : \kappa_v \rangle$

$b$

$n : \kappa'_n ?(x).P$

$P \{ v : \kappa_v ; a! \kappa_n ; b? \kappa'_n \}_x$
Provenance model

Provenance tracking

Old provenance of \( v \)

Sender and provenance of channel used

Receiver and provenance of channel used

\( a \)

\( n : \kappa_n \mid \langle v : \kappa_v \rangle \)

\( b \)

\( n : \kappa'_n ?(x).P \)

\( P\{v : \kappa_v; a!\kappa_n; b?\kappa'_n/x\} \)
Confidentiality in provenance systems

- Data may be public, yet its provenance confidential, or vice versa
- Principals who may access data are not necessarily the same as those who may access its provenance
- In general, fine grained access control over provenance “histories” is needed as different parts of it have different sensitivity
Hiding provenance trees
Example: photography competition

entry: ε, c!κ, entry: ε, c!κ, j?κ, j!κ
score: ε, j!κ, c: Contestant
a: Administrator
j: Judge

entry: ε, c!κ, a?κ, a!κ
score: ε, j!κ
entry: ε, c!κ, a?κ, a!κ, j?κ, j!κ
score: ε, j!κ
entry: ε, c!κ, a?κ, a!κ, j?κ, j!κ
score: ε, j!κ
Hiding provenance trees
Example: photography competition

entry: ε!κ
score: ε!κ_n

entry: ε!κ; a!κ'_n; j!κ'_n
score: ε, j!κ'_p

entry: ε!κ; a!κ'_n; j!κ'_n

entry: ε!κ; a!κ'_n; j!κ'_p

Hidden from j
Hidden from c
Confidentiality in provenance systems
a promising approach

✧ One value, multiple views

✧ Different principals have different views of the same provenance list based on their privileges

\[ entry : \varepsilon; c!\kappa_s; a?\kappa'_s; a!\kappa'_r; j?\kappa'_r; j!\kappa''_n; a?\kappa'_n; a!\kappa'_m \]
Confidentiality in provenance systems
a promising approach

✧ One value, multiple **views**

✧ Different principals have different views of the same provenance list based on their privileges

\[ entry : ε, c!κ_s; a?κ'_s; a!κ'_r; j?κ_r''; j!κ_n''; a?κ'_n; a!κ'_m \]
Confidentiality in provenance systems
a promising approach

❖ One value, multiple **views**

❖ Different principals have different views of the same provenance list based on their privileges

\[
entry : \varepsilon; c! \kappa_s; a? \kappa'_s; a! \kappa'_r; j? \kappa''_r; j! \kappa''_n; a? \kappa'_n; a! \kappa'_m
\]
Confidentiality in provenance systems
a promising approach

✧ One value, multiple views

✧ Different principals have different views of the same provenance list based on their privileges

\[
\text{entry} : \varepsilon; c!\kappa_s; a?\kappa'_s; a!\kappa'_r; j?\kappa''_r; j!\kappa''_n; a?\kappa'_n; a!\kappa'_m
\]
Confidentiality in provenance systems
a promising approach

⇧ One value, multiple views

⇧ Different principals have different views of the same provenance list based on their privileges

entry: \( \epsilon; c!\kappa_s; a'?\kappa'_s; a!\kappa_f; j'?\kappa''_r; j!\kappa''_n; a'?\kappa'_n; a!\kappa'_m \)

\( a \) \hspace{1cm} \( c \) \hspace{1cm} \( j \)
To achieve this, we use **groups**

- Different principals belong to different groups
- Group membership determines what parts of a provenance list a principal has access to

**Principals**

- Can create new groups: `new G`
- Can add other principals to their groups: `add(a,G)`
- Can **restrict** access to particular parts of a provenance tree to a particular group: `hide(v : κ, G)`

Confidentiality in provenance systems

a promising approach
Hiding provenance trees
photography competition

Group G

Group G’

Group membership at start of competition
Hiding provenance trees
photography competition

Group G

Group G’

Group membership at end of competition
Current work

- Correctness of provenance tracking: the provenance information determines the history of each piece of data accurately “enough”
  - Express this as a form of testing (on traces):
    \[
    \forall S (\forall t \in \llbracket S \rrbracket (t \to^* v : \kappa \implies \forall s \in \llbracket v : \kappa \rrbracket (s | t \to^* \checkmark)))
    \]
- Using provenance:
  - Provenance queries vs pattern restricted input
  - Trust in quality of data based on trust in principals and provenance of data
- Policies and types