CoCon

Andrei Popescu

Middlesex University London
Technical University Munich

Joint work with
Peter Lammich, Sergey Grebenshchikov,
Sudeep Kanav and Thomas Bauereiß
A Bit About My Whereabouts

Bucharest, Romania

Urbana-Champaign, US

Munich, Germany

Middlesex, UK

Interactive Theorem Proving

Isabelle
Conference system (think EasyChair, HotCRP) = web application for submitting and reviewing scientific papers
Conference system (think EasyChair, HotCRP) =
web application for submitting and reviewing scientific papers
How can it “go wrong”?
A Verified Conference System

Conference system (think EasyChair, HotCRP) =
web application for submitting and reviewing scientific papers
How can it “go wrong”?

It is our pleasure to inform you that your paper has been accepted to the IEEE Symposium of Security and Privacy (Oakland) 2012.

We are sorry to inform you that your paper was not accepted for this year’s conference.
We apologize for an earlier “acceptance” notification. It was due to a system error.

We received 307 submissions and only accepted 40 of them; the selection criteria were very stringent.

Implemented and verified CoCon, a conference system with confidentiality guarantees
Conference system (think EasyChair, HotCRP) = web application for submitting and reviewing scientific papers
How can it “go wrong”? 

It is our pleasure to inform you that your paper has been accepted to the IEEE Symposium of Security and Privacy (Oakland) 2012.
We are sorry to inform you that your paper was not accepted for this year’s conference. We apologize for an earlier “acceptance” notification. It was due to a system error.
Conference system (think EasyChair, HotCRP) = web application for submitting and reviewing scientific papers

How can it “go wrong”? 

It is our pleasure to inform you that your paper has been accepted to the IEEE Symposium of Security and Privacy (Oakland) 2012.

We are sorry to inform you that your paper was not accepted for this year’s conference. We apologize for an earlier “acceptance” notification. It was due to a system error. We received 307 submissions and only accepted 40 of them; the selection criteria were very stringent.
A Verified Conference System

Conference system (think EasyChair, HotCRP) =
web application for submitting and reviewing scientific papers
How can it “go wrong”?

It is our pleasure to inform you that your paper has been accepted to the IEEE Symposium of Security and Privacy (Oakland) 2012.
We are sorry to inform you that your paper was not accepted for this year’s conference. We apologize for an earlier “acceptance” notification. It was due to a system error. We received 307 submissions and only accepted 40 of them; the selection criteria were very stringent.

WARNING: HotCRP version 2.47 (commit range 94ca5a0e43bd7dd0565c2c8dc7d8f710a206ab49 through 9c1b45475411ecb85d46bad1f76064881792b038) was subject to an information exposure where some authors could see PC comments. Users of affected versions should upgrade or set the following option in Code/options.inc: $Opt["disableCapabilities"] = true;
A Verified Conference System

Conference system (think EasyChair, HotCRP) =
web application for submitting and reviewing scientific papers
How can it “go wrong”?

It is our pleasure to inform you that your paper has been accepted to the IEEE
Symposium of Security and Privacy (Oakland) 2012.
We are sorry to inform you that your paper was not accepted for this year’s conference.
We apologize for an earlier “acceptance” notification. It was due to a system error. We
received 307 submissions and only accepted 40 of them; the selection criteria were very
stringent.

WARNING: HotCRP version 2.47 (commit range 94ca5a0e43bd7dd0565c2c8dc7d8f710a206ab49 through
9c1b45475411ecb85d46badf1f76064881792b038) was subject to an information exposure where some
authors could see PC comments. Users of affected versions should upgrade or set the following option in
Code/options.inc: $Opt["disableCapabilities"] = true;

Implemented and verified CoCon, a conference system with confidentiality guarantees
CoCon = Feature-Rich Conference Management System (similar to EasyChair and HotCRP)

https://cocon.in.tum.de
CoCon’s Architecture

Isabelle Specification

---

code generation

Scala Program

Scalatra API

Web Application
CoCon’s Formal Guarantees

- A user learns nothing about a paper’s content beyond the last submitted version unless she becomes an author of the paper.
CoCon’s Formal Guarantees

- A user learns nothing about a paper’s content beyond the last submitted version unless she becomes an author of the paper.
- A user learns nothing about a paper’s assigned reviewers beyond the fact that they are PC members unless he becomes a non-conflict PC member.
CoCon’s Formal Guarantees

- A user learns nothing about a paper’s content beyond the last submitted version unless she becomes an author of the paper.
- A user learns nothing about a paper’s assigned reviewers beyond the fact that they are PC members unless he becomes a non-conflict PC member.

Source
CoCon’s Formal Guarantees

- A user learns nothing about a paper’s content beyond the last submitted version unless she becomes an author of the paper.
- A user learns nothing about a paper’s assigned reviewers beyond the fact that they are PC members unless he becomes a non-conflict PC member.

Source  Sink
A user learns nothing about a paper’s content beyond the last submitted version unless she becomes an author of the paper.

A user learns nothing about a paper’s assigned reviewers beyond the fact that they are PC members unless he becomes a non-conflict PC member.
CoCon’s Formal Guarantees

- A user learns nothing about a paper’s content beyond the last submitted version unless she becomes an author of the paper.
- A user learns nothing about a paper’s assigned reviewers beyond the fact that they are PC members unless he becomes a non-conflict PC member.

Source  Sink  Trigger  Bound
CoCon’s Formal Guarantees

- A user learns nothing about a paper’s content beyond the last submitted version unless she becomes an author of the paper.
- A user learns nothing about a paper’s assigned reviewers beyond the fact that they are PC members unless he becomes a non-conflict PC member.

Source  Sink  Trigger  Bound

Bounded-Deducibility (BD) Security
CoCon’s Formal Guarantees

- A user learns nothing about a paper’s content beyond the last submitted version unless she becomes an author of the paper.
- A user learns nothing about a paper’s assigned reviewers beyond the fact that they are PC members unless he becomes a non-conflict PC member.

Bounded-Deducibility (BD) Security
- Applicable to arbitrary I/O automata.
CoCon’s Formal Guarantees

- A user learns nothing about a paper’s content beyond the last submitted version unless she becomes an author of the paper.
- A user learns nothing about a paper’s assigned reviewers beyond the fact that they are PC members unless he becomes a non-conflict PC member.

Source  Sink  Trigger  Bound

Bounded-Deducibility (BD) Security
- applicable to arbitrary I/O automata
- capable of expressing flexible declassification bounds and triggers
SystemTrace \subseteq \text{Event}^{*}

\text{SystemTrace} \quad \text{Secret}^{*}

\text{Obs}^{*}
SystemTrace $\subseteq$ Event$^*$

$S = \text{produce secrets from selected events}$

SystemTrace $\xrightarrow{S} \text{Secret}^*$

Obs$^*$
SystemTrace ⊆ Event∗

O = produce observations from selected events

SystemTrace → S → Secret∗

O → Obs∗
SystemTrace ⊆ Event*  

\[ T : \text{Event} \rightarrow \text{Bool} \]

\[ \neg T \quad \text{SystemTrace} \quad S \quad \text{Secret}* \]

\[ O \rightarrow \text{Obs}* \]
SystemTrace \subseteq \text{Event}^* \\

T : \text{Event} \rightarrow \text{Bool} \quad B \text{ relation on Secret}^* \\

\neg T \quad \text{SystemTrace} \quad S \quad \text{Secret}^* \quad B \\

O \quad \text{Obs}^*
SystemTrace \subseteq \text{Event}^* \\

Unless T occurs, O can learn nothing about S beyond B
Bounded-Deducibility (BD) Security

SystemTrace ⊆ Event*

Unless T occurs, O can learn nothing about S beyond B

\[ \neg T \quad \text{SystemTrace} \quad S \quad \text{Secret}^* \quad B \]
SystemTrace ⊆ Event*

Unless T occurs, O can learn nothing about S beyond B

Diagram:

- T
- SystemTrace → S → Secret*
- O
- Obs* → o, o
SystemTrace ⊆ Event*

Unless T occurs, O can learn nothing about S beyond B

Diagram:

\[ \neg T \xrightarrow{\text{SystemTrace}} S \xrightarrow{\text{Secret*}} B \]

O

Obs* o o
SystemTrace $\subseteq$ Event$^*$

Unless $T$ occurs, $O$ can learn nothing about $S$ beyond $B$

\[ \neg T \quad \text{SystemTrace} \quad S \quad \text{Secret}^* \quad B \]

\[ s 
\quad s 
\quad s 
\quad s 
\]

\[ o 
\quad o \]
SystemTrace $\subseteq$ Event$^*$

Unless $T$ occurs, $O$ can learn nothing about $S$ beyond $B$

$\neg T$  SystemTrace  $S$  Secret$^*$  $B$

O  

Obs$^*$  $O$  $O$

$s$  $s$  $B$  $s$  $s$
SystemTrace ⊆ Event∗

Unless T occurs, O can learn nothing about S beyond B

\[
\begin{align*}
\neg T & \quad \text{SystemTrace} & \quad S & \quad \text{Secret∗} & \quad B \\
O & \quad \text{Obs∗} & \quad O & \quad o & \quad o
\end{align*}
\]
Bounded-Deducibility (BD) Security

SystemTrace $\subseteq$ Event$^*$

Unless $T$ occurs, $O$ can learn nothing about $S$ beyond $B$

\[ O \xrightarrow{\text{Obs}^*} 00 \quad 00 \]
Bounded-Deducibility (BD) Security

Proof by unwinding

$\neg T$ → SystemTrace → $S$ → Secret* → B

O → Obs*
Proof by unwinding
Action

¬ T  SystemTrace  S  Secret*  B

O

Obs*  o
Bounded-Deducibility (BD) Security

Proof by unwinding
Action / Reaction: Match

¬ T  SystemTrace  S  Secret*  B

O

Obs*  o o
Bounded-Deducibility (BD) Security

Proof by unwinding
Action

\[\neg T \quad \text{SystemTrace} \quad S \quad \text{Secret}^* \quad B\]

\[\text{O} \quad \text{O}\]

\[\text{Obs}^* \quad o \quad o\]
Bounded-Deducibility (BD) Security

Proof by unwinding
Action / Reaction: Ignore

¬ T SystemTrace S Secret* B

O Obs* o o

\[\neg T \rightarrow \text{SystemTrace} \rightarrow S \rightarrow \text{Secret}^* \rightarrow B\]
Bounded-Deducibility (BD) Security

Proof by unwinding
Action

\[ \neg T \rightarrow \text{SystemTrace} \rightarrow S \rightarrow \text{Secret}^* \rightarrow B \]

O

Obs*  oo o
Bounded-Deducibility (BD) Security

Proof by unwinding
Action / Reaction: Match

¬ T SystemTrace S Secret*
O Obs* 00 00

B s
Proof by unwinding
Independent action …

Bounded-Deducibility (BD) Security
Bounded-Deducibility (BD) Security

Proof by unwinding
Independent action

\[ \neg T \quad \text{SystemTrace} \quad S \quad \text{Secret*} \quad B \]

O

Obs*  oo  oo
Bounded-Deducibility (BD) Security

Proof by unwinding

Independent action

\[ \neg T \quad \text{SystemTrace} \quad S \quad \text{Secret}^* \quad B \]

\[ \text{Obs}^* \quad 00 \quad 00 \]

\[ \text{O} \]

\[ \text{B} \]
Bounded-Deducibility (BD) Security

Proof by unwinding

\( B \mapsto \Delta \subseteq \text{State} \times \text{Secret} \times \text{State} \times \text{Secret} \times \text{SystemTrace} \)

\( \neg T \xrightarrow{\text{SystemTrace}} S \xrightarrow{\text{Secret}*} B \)

O

\( \text{Obs}^* \quad o o \\
\text{O} \)

\( \text{O} \)
Bounded-Deducibility (BD) Security

Proof by unwinding

\[ B \mapsto \Delta \subseteq \text{State} \times \text{Secret}^* \times \text{State} \times \text{Secret}^* \]

\[ \neg T \quad \text{SystemTrace} \quad S \quad \rightarrow \quad \text{Secret}^* \quad B \]

\[ \text{Obs}^* \quad 00 \quad 00 \]
Proof by Unwinding

\[ \Delta \subseteq \text{State} \times \text{Secret}^* \times \text{State} \times \text{Secret}^* + \]

Strategy for:
- when to act independently
- when to react
- if react: when to match and when to ignore
\[ \Delta \subseteq \text{State} \times \text{Secret}^* \times \text{State} \times \text{Secret}^* \]
\[ \Delta \subseteq \text{State} \times \text{Secret}^* \times \text{State} \times \text{Secret}^* \]

Strategy for:
- when to act independently
- when to react
- if react: when to match and when to ignore
<table>
<thead>
<tr>
<th>Source</th>
<th>Trigger</th>
<th>Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paper Content</td>
<td>Paper Authorship</td>
<td>Last Uploaded Version</td>
</tr>
<tr>
<td></td>
<td>Paper Authorship or PC Membership&lt;sup&gt;B&lt;/sup&gt;</td>
<td>Nothing</td>
</tr>
<tr>
<td>Review</td>
<td>Review Authorship</td>
<td>Last Edited Version</td>
</tr>
<tr>
<td></td>
<td>Review Authorship or Non-Conflict PC Membership&lt;sup&gt;D&lt;/sup&gt;</td>
<td>Before Discussion and All the Later Versions</td>
</tr>
<tr>
<td></td>
<td>Review Authorship or Non-Conflict PC Membership&lt;sup&gt;D&lt;/sup&gt; or PC Membership&lt;sup&gt;N&lt;/sup&gt; or Paper Authorship&lt;sup&gt;N&lt;/sup&gt;</td>
<td>Nothing</td>
</tr>
<tr>
<td>Discussion</td>
<td>Non-Conflict PC Membership</td>
<td>Nothing</td>
</tr>
<tr>
<td>Decision</td>
<td>Non-Conflict PC Membership</td>
<td>Last Edited Version</td>
</tr>
<tr>
<td></td>
<td>Non-Conflict PC Membership or PC Membership&lt;sup&gt;N&lt;/sup&gt; or Paper Authorship&lt;sup&gt;N&lt;/sup&gt;</td>
<td>Nothing</td>
</tr>
<tr>
<td>Reviewer Assignment to Paper</td>
<td>Non-Conflict PC Membership&lt;sup&gt;R&lt;/sup&gt;</td>
<td>Non-Conflict PC Membership of Reviewers</td>
</tr>
</tbody>
</table>

Phase Stamps:  

- B = Bidding
- D = Discussion
- N = Notification
- R = Review
<table>
<thead>
<tr>
<th>Source</th>
<th>Trigger</th>
<th>Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paper Content</td>
<td>Paper Authorship</td>
<td>Last Uploaded Version</td>
</tr>
<tr>
<td></td>
<td>Paper Authorship or PC Membership(^B)</td>
<td>Nothing</td>
</tr>
<tr>
<td>Review</td>
<td>Review Authorship</td>
<td>Last Edited Version</td>
</tr>
<tr>
<td></td>
<td>Review Authorship or Non-Conflict PC Membership(^D)</td>
<td>Before Discussion and All the Later Versions</td>
</tr>
<tr>
<td></td>
<td>Review Authorship or Non-Conflict PC Membership(^D) or PC Membership(^N) or Paper Authorship(^N)</td>
<td>Last Edited Version Before Notification</td>
</tr>
<tr>
<td></td>
<td>Non-Conflict PC Membership</td>
<td>Nothing</td>
</tr>
<tr>
<td>Discussion</td>
<td>Non-Conflict PC Membership</td>
<td>Nothing</td>
</tr>
<tr>
<td>Decision</td>
<td>Non-Conflict PC Membership</td>
<td>Last Edited Version</td>
</tr>
<tr>
<td></td>
<td>Non-Conflict PC Membership or PC Membership(^N) or Paper Authorship(^N)</td>
<td>Nothing</td>
</tr>
<tr>
<td>Reviewer Assignment to Paper</td>
<td>Non-Conflict PC Membership(^R)</td>
<td>Non-Conflict PC Membership of Reviewers</td>
</tr>
</tbody>
</table>

Phase Stamps: B = Bidding, D = Discussion, N = Notification, R = Review
<table>
<thead>
<tr>
<th>Source</th>
<th>Trigger</th>
<th>Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paper Content</td>
<td>Paper Authorship</td>
<td>Last Uploaded Version</td>
</tr>
<tr>
<td></td>
<td>Paper Authorship or PC Membership^B</td>
<td>Nothing</td>
</tr>
<tr>
<td>Review</td>
<td>Review Authorship</td>
<td>Last Edited Version</td>
</tr>
<tr>
<td></td>
<td>Review Authorship or Non-Conflict PC Membership^D</td>
<td>Before Discussion and All the Later Versions</td>
</tr>
<tr>
<td></td>
<td>Review Authorship or Non-Conflict PC Membership^D or PC Membership^N or Paper Authorship^N</td>
<td>Nothing</td>
</tr>
<tr>
<td>Discussion</td>
<td>Non-Conflict PC Membership</td>
<td>Nothing</td>
</tr>
<tr>
<td>Decision</td>
<td>Non-Conflict PC Membership</td>
<td>Last Edited Version</td>
</tr>
<tr>
<td></td>
<td>Non-Conflict PC Membership or PC Membership^N or Paper Authorship^N</td>
<td>Nothing</td>
</tr>
<tr>
<td>Reviewer Assignment to Paper</td>
<td>Non-Conflict PC Membership^R</td>
<td>Non-Conflict PC Membership of Reviewers</td>
</tr>
</tbody>
</table>

Phase Stamps: B = Bidding, D = Discussion, N = Notification, R = Review
<table>
<thead>
<tr>
<th>Source</th>
<th>Trigger</th>
<th>Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paper Content</td>
<td>Paper Authorship</td>
<td>Last Uploaded Version</td>
</tr>
<tr>
<td></td>
<td>Paper Authorship or PC Membership(^B)</td>
<td>Nothing</td>
</tr>
<tr>
<td>Review</td>
<td>Review Authorship</td>
<td>Last Edited Version Before Discussion and All the Later Versions</td>
</tr>
<tr>
<td></td>
<td>Review Authorship or Non-Conflict PC Membership(^D)</td>
<td>Last Edited Version Before Notification</td>
</tr>
<tr>
<td></td>
<td>Review Authorship or Non-Conflict PC Membership(^D) or PC Membership(^N) or Paper Authorship(^N)</td>
<td>Nothing</td>
</tr>
<tr>
<td>Discussion</td>
<td>Non-Conflict PC Membership</td>
<td>Nothing</td>
</tr>
<tr>
<td>Decision</td>
<td>Non-Conflict PC Membership</td>
<td>Last Edited Version</td>
</tr>
<tr>
<td></td>
<td>Non-Conflict PC Membership or PC Membership(^N) or Paper Authorship(^N)</td>
<td>Nothing</td>
</tr>
<tr>
<td>Reviewer Assignment to Paper</td>
<td>Non-Conflict PC Membership(^R)</td>
<td>Non-Conflict PC Membership of Reviewers</td>
</tr>
</tbody>
</table>

Phase Stamps:  \(B = \text{Bidding}, \ D = \text{Discussion}, \ N = \text{Notification}, \ R = \text{Review}\)
<table>
<thead>
<tr>
<th>Source</th>
<th>Trigger</th>
<th>Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paper Content</td>
<td>Paper Authorship</td>
<td>Last Uploaded Version</td>
</tr>
<tr>
<td></td>
<td>Paper Authorship or PC Membership&lt;sup&gt;B&lt;/sup&gt;</td>
<td>Nothing</td>
</tr>
<tr>
<td>Review</td>
<td>Review Authorship</td>
<td>Last Edited Version Before Discussion and All the Later Versions</td>
</tr>
<tr>
<td></td>
<td>Review Authorship or Non-Conflict PC Membership&lt;sup&gt;D&lt;/sup&gt;</td>
<td>Last Edited Version Before Notification</td>
</tr>
<tr>
<td></td>
<td>Review Authorship or Non-Conflict PC Membership&lt;sup&gt;D&lt;/sup&gt; or PC Membership&lt;sup&gt;N&lt;/sup&gt; or Paper Authorship&lt;sup&gt;N&lt;/sup&gt;</td>
<td>Nothing</td>
</tr>
<tr>
<td>Discussion</td>
<td>Non-Conflict PC Membership</td>
<td>Nothing</td>
</tr>
<tr>
<td>Decision</td>
<td>Non-Conflict PC Membership</td>
<td>Last Edited Version</td>
</tr>
<tr>
<td></td>
<td>Non-Conflict PC Membership or PC Membership&lt;sup&gt;N&lt;/sup&gt; or Paper Authorship&lt;sup&gt;N&lt;/sup&gt;</td>
<td>Nothing</td>
</tr>
<tr>
<td>Reviewer Assignment to Paper</td>
<td>Non-Conflict PC Membership&lt;sup&gt;R&lt;/sup&gt;</td>
<td>Non-Conflict PC Membership of Reviewers</td>
</tr>
</tbody>
</table>

Phase Stamps: B = Bidding, D = Discussion, N = Notification, R = Review
<table>
<thead>
<tr>
<th>Source</th>
<th>Trigger</th>
<th>Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paper Content</td>
<td>Paper Authorship</td>
<td>Last Uploaded Version</td>
</tr>
<tr>
<td></td>
<td>Paper Authorship or PC Membership&lt;sup&gt;B&lt;/sup&gt;</td>
<td>Nothing</td>
</tr>
<tr>
<td>Review</td>
<td>Review Authorship</td>
<td>Last Edited Version Before Discussion and All the Later Versions</td>
</tr>
<tr>
<td></td>
<td>Review Authorship or Non-Conflict PC Membership&lt;sup&gt;D&lt;/sup&gt;</td>
<td>Last Edited Version Before Notification</td>
</tr>
<tr>
<td></td>
<td>Review Authorship or Non-Conflict PC Membership&lt;sup&gt;D&lt;/sup&gt; or PC Membership&lt;sup&gt;N&lt;/sup&gt; or Paper Authorship&lt;sup&gt;N&lt;/sup&gt;</td>
<td>Nothing</td>
</tr>
<tr>
<td>Discussion</td>
<td>Non-Conflict PC Membership</td>
<td>Nothing</td>
</tr>
<tr>
<td>Decision</td>
<td>Non-Conflict PC Membership</td>
<td>Last Edited Version</td>
</tr>
<tr>
<td></td>
<td>Non-Conflict PC Membership or PC Membership&lt;sup&gt;N&lt;/sup&gt; or Paper Authorship&lt;sup&gt;N&lt;/sup&gt;</td>
<td>Nothing</td>
</tr>
<tr>
<td>Reviewer Assignment to Paper</td>
<td>Non-Conflict PC Membership&lt;sup&gt;R&lt;/sup&gt;</td>
<td>Non-Conflict PC Membership of Reviewers</td>
</tr>
</tbody>
</table>

Phase Stamps:  
B = Bidding,  
D = Discussion,  
N = Notification,  
R = Review
<table>
<thead>
<tr>
<th>Source</th>
<th>Trigger</th>
<th>Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paper Content</td>
<td>Paper Authorship</td>
<td>Last Uploaded Version</td>
</tr>
<tr>
<td></td>
<td>Paper Authorship or PC Membership(_B)</td>
<td>Nothing</td>
</tr>
<tr>
<td>Review</td>
<td>Review Authorship</td>
<td>Last Edited Version</td>
</tr>
<tr>
<td></td>
<td>Review Authorship or Non-Conflict PC Membership(_D)</td>
<td>Before Discussion and All the Later Versions</td>
</tr>
<tr>
<td></td>
<td>Review Authorship or Non-Conflict PC Membership(_D) or PC Membership(_N) or Paper Authorship(_N)</td>
<td>Last Edited Version Before Notification</td>
</tr>
<tr>
<td></td>
<td>Non-Conflict PC Membership</td>
<td>Nothing</td>
</tr>
<tr>
<td>Discussion</td>
<td>Non-Conflict PC Membership</td>
<td>Nothing</td>
</tr>
<tr>
<td>Decision</td>
<td>Non-Conflict PC Membership</td>
<td>Last Edited Version</td>
</tr>
<tr>
<td></td>
<td>Non-Conflict PC Membership or PC Membership(_N) or Paper Authorship(_N)</td>
<td>Nothing</td>
</tr>
<tr>
<td>Reviewer Assignment to Paper</td>
<td>Non-Conflict PC Membership(_R)</td>
<td>Non-Conflict PC Membership of Reviewers</td>
</tr>
</tbody>
</table>

Phase Stamps:  \(B = \) Bidding,  \(D = \) Discussion,  \(N = \) Notification,  \(R = \) Review
<table>
<thead>
<tr>
<th>Source</th>
<th>Trigger</th>
<th>Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paper Content</td>
<td>Paper Authorship</td>
<td>Last Uploaded Version</td>
</tr>
<tr>
<td></td>
<td>Paper Authorship or PC Membership⁸</td>
<td>Nothing</td>
</tr>
<tr>
<td>Review</td>
<td>Review Authorship</td>
<td>Last Edited Version Before Discussion and All the Later Versions</td>
</tr>
<tr>
<td></td>
<td>Review Authorship or Non-Conflict PC Membership⁸</td>
<td>Last Edited Version Before Notification</td>
</tr>
<tr>
<td></td>
<td>Non-Conflict PC Membership⁸ or PC Membership⁹ or Paper Authorship⁹</td>
<td>Nothing</td>
</tr>
<tr>
<td>Discussion</td>
<td>Non-Conflict PC Membership</td>
<td>Nothing</td>
</tr>
<tr>
<td>Decision</td>
<td>Non-Conflict PC Membership</td>
<td>Last Edited Version</td>
</tr>
<tr>
<td></td>
<td>Non-Conflict PC Membership or PC Membership⁹ or Paper Authorship⁹</td>
<td>Nothing</td>
</tr>
<tr>
<td>Reviewer Assignment to Paper</td>
<td>Non-Conflict PC Membership⁸</td>
<td>Non-Conflict PC Membership of Reviewers and Number of Reviewers</td>
</tr>
</tbody>
</table>

Phase Stamps: B = Bidding, D = Discussion, N = Notification, R = Review
<table>
<thead>
<tr>
<th>Source</th>
<th>Trigger</th>
<th>Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paper Content</td>
<td>Paper Authorship</td>
<td>Last Uploaded Version</td>
</tr>
<tr>
<td></td>
<td>Paper Authorship or PC Membership(^B)</td>
<td>Nothing</td>
</tr>
<tr>
<td>Review</td>
<td>Review Authorship</td>
<td>Last Edited Version</td>
</tr>
<tr>
<td></td>
<td>Review Authorship or Non-Conflict PC Membership(^D)</td>
<td>Before Discussion and All the Later Versions</td>
</tr>
<tr>
<td></td>
<td>Review Authorship or Non-Conflict PC Membership(^D) or PC Membership(^N) or Paper Authorship(^N)</td>
<td>Last Edited Version Before Notification</td>
</tr>
<tr>
<td></td>
<td>Review Authorship or Non-Conflict PC Membership(^D) or PC Membership(^N) or Paper Authorship(^N)</td>
<td>Nothing</td>
</tr>
<tr>
<td>Discussion</td>
<td>Non-Conflict PC Membership</td>
<td>Nothing</td>
</tr>
<tr>
<td>Decision</td>
<td>Non-Conflict PC Membership</td>
<td>Last Edited Version</td>
</tr>
<tr>
<td></td>
<td>Non-Conflict PC Membership or PC Member(^N) or Paper Authorship(^N)</td>
<td>Nothing</td>
</tr>
<tr>
<td>Reviewer Assignment</td>
<td>Non-Conflict PC Membership(^R)</td>
<td>Non-Conflict PC Membership of Reviewers and Number of Reviewers</td>
</tr>
<tr>
<td>to Paper</td>
<td>Non-Conflict PC Membership(^R) or Paper Authorship(^N)</td>
<td>Non-Conflict PC Membership of Reviewers</td>
</tr>
</tbody>
</table>

Phase Stamps:  
B = Bidding,  
D = Discussion,  
N = Notification,  
R = Review
Overall Verification Effort

BD Security Framework: 1000 LOC

Confidentiality properties: 5000 LOC

Safety verification: 1000 LOC

Accountability: 700 LOC
Overall Verification Effort

BD Security Framework: 1000 LOC

Confidentiality properties: 5000 LOC
1. Only non-conflict PC members may learn such and such

Safety verification: 1000 LOC

Accountability: 700 LOC
BD Security Framework: 1000 LOC

Confidentiality properties: 5000 LOC
1. Only non-conflict PC members may learn such and such

Safety verification: 1000 LOC
2. And authors are always in conflict with their papers

Accountability: 700 LOC
Overall Verification Effort

BD Security Framework: 1000 LOC

Confidentiality properties: 5000 LOC
1. Only non-conflict PC members may learn such and such

Safety verification: 1000 LOC
2. And authors are always in conflict with their papers
(1) + (2) $\rightarrow$ Authors never learn such and such

Accountability: 700 LOC
Overall Verification Effort

BD Security Framework: 1000 LOC

Confidentiality properties: 5000 LOC

Safety verification: 1000 LOC

Accountability: 700 LOC
We proved: one cannot learn beyond such an such unless one is or becomes such and such
We proved: one cannot learn beyond such an such unless one is or becomes such and such
But how can one become such and such?
E.g., how could person become an author of document?
We proved: one cannot learn beyond such an such unless one is or becomes such and such

But how can one become such and such?
E.g., how could  ⬀ become an author of ⚪️ ⬔?

까요  ⬀ has registered ⚪️
We proved: one cannot learn beyond such an such unless one is or becomes such and such.

But how can one become such and such?
E.g., how could 🤖 become an author of 📄?

 inversión has registered 📄
We proved: one cannot learn beyond such an such unless one is or becomes such and such

But how can one become such and such?
E.g., how could \(\text{\includegraphics[width=1.5cm]{person}}\) become an author of \(\text{\includegraphics[width=1.5cm]{file}}\) ?

\(\text{\includegraphics[width=1.5cm]{person}}\) has registered \(\text{\includegraphics[width=1.5cm]{file}}\)

\(\ldots\)

\(\text{\includegraphics[width=1.5cm]{person}}\)
We proved: one cannot learn beyond such an such unless one is or becomes such and such

But how can one become such and such?
E.g., how could \[\text{\includegraphics[width=0.05\textwidth]{person}}\] become an author of \[\text{\includegraphics[width=0.05\textwidth]{book}}\]?

\[\text{\includegraphics[width=0.05\textwidth]{person}} \quad \text{has registered} \quad \text{\includegraphics[width=0.05\textwidth]{book}}
\]

\[\downarrow\]

\[\vdots\]

\[\downarrow\]

\[\text{\includegraphics[width=0.05\textwidth]{person}}\]

Confidentiality + Safety + Accountability \[\longrightarrow\] Relax
• Generic parameterized security notion
• Associated unwinding proof method
• Instantiated to reason about CoCon’s confidentiality:

  What, when, by whom can be learned about the documents in the system (papers, reviews, discussions, reviewer assignment)
Summary of Verification

- Generic parameterized security notion
- Associated unwinding proof method
- Instantiated to reason about CoCon’s confidentiality:

  What, when, by whom
can be learned about
  the documents in the system
  (papers, reviews, discussions, reviewer assignment)

Kanav, Lammich, Popescu. A conference management system with verified document confidentiality. CAV 2014

“The authors cunningly chose a topic that directly speaks to the reviewers of their paper.”
CoCon, a *concrete* system, has inspired BD security, a *very abstract* framework...
What I Like About CoCon

CoCon, a **concrete** system, has inspired BD security, a **very abstract** framework.

The size of the proof goals arising from CoCon have **compelled** us to improve compositionality and automation of BD security.
CoCon, a **concrete** system, has inspired BD security, a **very abstract** framework.

The size of the proof goals arising from CoCon have **compelled** us to improve compositionality and automation of BD security.

And practical considerations about CoCon keep bringing new theoretical challenges...
CoCon at TABLEAUX 2015?

Hans de Nivelle

conference chair

CoCon? Not a chance.

But maybe CoCon++...
CoCon at TABLEAUX 2015?

Hans de Nivelle
conference chair
CoCon at TABLEAUX 2015?

Hans de Nivelle
conference chair

CoCon? Not a chance.
CoCon at TABLEAUX 2015?

CoCon? Not a chance.

But maybe CoCon++...
CoCon plus the following features:
CoCon++

Is CoCon++ as (provably) secure as CoCon?

CoCon plus the following features:

Various convenience listings, e.g.:
  For PC members: papers listed by average score
  For the chair: paper load of each PC member, reviewer number for each paper
  …
CoCon plus the following features:

Various convenience listings, e.g.:

For PC members: papers listed by average score
For the chair: paper load of each PC member, reviewer number for each paper

Email notifications:

To authors about the decision
To PC members about the addition of comments or new reviews to the papers under their review
Is CoCon++ as (provably) secure as CoCon?

CoCon plus the following features:

Various convenience listings, e.g.:
- For PC members: papers listed by average score
- For the chair: paper load of each PC member, reviewer number for each paper
- ...

Email notifications:
- To authors about the decision
- To PC members about the addition of comments or new reviews to the papers under their review
- ...

Is CoCon++ as (provably) secure as CoCon?

CoCon plus the following features:

Various convenience listings, e.g.: OK

- For PC members: papers listed by average score
- For the chair: paper load of each PC member, reviewer number for each paper

Email notifications:

- To authors about the decision
- To PC members about the addition of comments or new reviews to the papers under their review
Is CoCon++ as (provably) secure as CoCon?

CoCon plus the following features:

**Various convenience listings, e.g.: OK**
- For PC members: papers listed by average score
- For the chair: paper load of each PC member, reviewer number for each paper
  ...

**Email notifications:** OK, if done right
- To authors about the decision
- To PC members about the addition of comments or new reviews to the papers under their review
  ...

Various convenience listings, e.g.:

For PC members: papers listed by average score
Various convenience listings, e.g.:

For PC members: papers listed by average score

Had proved about CoCon’s kernel:
An author learns nothing about the score of her paper before the Notification phase.
Various convenience listings, e.g.:

For PC members: papers listed by average score

Had proved about CoCon’s kernel:
An author learns nothing about the score of her paper before the Notification phase.
Various convenience listings, e.g.:

For PC members: papers listed by average score

Had proved about CoCon’s kernel:
An author learns nothing about the score of her paper before the Notification phase.

I’m also an author and I see my paper listed somewhere in the middle.
Various convenience listings, e.g.:

For PC members: papers listed by average score

Had proved about CoCon’s kernel:
An author learns nothing about the score of her paper before the Notification phase.

I’m also an author and I see my paper listed somewhere in the middle.
Does it mean I can already infer something about its score?
Various convenience listings, e.g.:

For PC members: papers listed by average score

Had proved about CoCon’s kernel:
An author learns nothing about the score of her paper before the Notification phase.

I’m also an author and I see my paper listed somewhere in the middle. Does it mean I can already infer something about its score?

Our answer: No, but we apologize for the confusing behavior.
Various convenience listings, e.g.:

For PC members: papers listed by average score

Had proved about CoCon’s kernel:
An author learns nothing about the score of her paper before the Notification phase.
Various convenience listings, e.g.:

For PC members: papers listed by average score

Had proved about CoCon’s kernel:
An author learns nothing about the score of her paper before the Notification phase.

Convenience features are written around the kernel invoking sequences of kernel actions. E.g.:

When listing papers to a user $U$, only invoke actions permitted to $U$
Various convenience listings, e.g.:

For PC members: papers listed by average score

Had proved about CoCon’s kernel:
An author learns nothing about the score of her paper before the Notification phase.

Convenience features are written around the kernel invoking sequences of kernel actions. E.g.:

When listing papers to a user $U$, only invoke actions permitted to $U$
If PC member is also author, can see content but not score
Various convenience listings, e.g.:

For PC members: papers listed by average score

Had proved about CoCon’s kernel:
An author learns nothing about the score of her paper before the Notification phase.

Convenience features are written around the kernel invoking sequences of kernel actions. E.g.:

When listing papers to a user $U$, only invoke actions permitted to $U$
If PC member is also author, can see content but not score
Previous confusion caused by “unknown” treated as 0
Email notifications:

To PC members about the addition of comments or new reviews to papers under their review
Email notifications:

To PC members about the addition of comments or new reviews to papers under their review
Email notifications:

   To PC members about the addition of comments or new reviews to any paper
Email notifications:

To PC members about the addition of comments or new reviews to any paper

Why am I notified when a new (confidential!) comment has been added to my paper?
Email notifications:

To PC members about the addition of comments or new reviews to any paper

Why am I notified when a new (confidential!) comment has been added to my paper?

To avoid this: To gather data for a notification to a user, only employ sequences of actions permitted to that user.
Given a confidentiality-verified system

Extend the system

- either with “convenience” features
- or with information sent on additional channels, but to the same parties
Given a confidentiality-verified system

Extend the system

- either with “convenience” features
- or with information sent on additional channels but to the same parties

Each newly added action:

First invokes sequences of old actions, each chosen based on the output of previous actions only
Given a confidentiality-verified system

Extend the system

- either with “convenience” features
- or with information sent on additional channels but to the same parties

Each newly added action:

First invokes sequences of old actions, each chosen based on the output of previous actions only

Then synthesizes output from these old actions’ outputs
Given a confidentiality-verified system

Extend the system

• either with “convenience” features
• or with information sent on additional channels but to the same parties

Each newly added action:

First invokes sequences of old actions, each chosen based on the output of previous actions only
Then synthesizes output from these old actions’ outputs

In particular, does not “peek” directly into the state!
Given a confidentiality-verified system

Extend the system

- either with “convenience” features
- or with information sent on additional channels but to the same parties

Each newly added action:

First invokes sequences of old actions, each chosen based on the output of previous actions only

Then synthesizes output from these old actions’ outputs

In particular, does not “peek” directly into the state!

Example for CoCon++...
For PC member $U$, during Discussion, add new action $\text{listPapersSorted}(U)$ that lists all papers sorted by score.
Example for CoCon++

For PC member $U$, during Discussion, add new action $\text{listPapersSorted}(U)$ that lists all papers sorted by score.

1. $\text{listPapers}(U)$: output $o_1 \ldots o_k$
For PC member $U$, during Discussion, add new action `listPapersSorted(U)` that lists all papers sorted by score.

1. `listPapers(U)`: output $o_1 \ldots o_k$

2. For each $i \leq k$, `checkConflict(U, o_i)`
For PC member $U$, during Discussion, add new action $\text{listPapersSorted}(U)$ that lists all papers sorted by score.

1. $\text{listPapers}(U)$: output $o_1 \ldots o_k$

2. For each $i \leq k$, $\text{checkConflict}(U, o_i)$

3. Based on 2, split in two lists:
   - Non-conflict: $p_1 \ldots p_m$
   - Conflict: $q_1 \ldots q_n$

Summary: old action invocation + some output filtering
Example for CoCon++

For PC member $U$, during Discussion, add new action $\text{listPapersSorted}(U)$ that lists all papers sorted by score.

1. $\text{listPapers}(U)$: output $o_1 \ldots o_k$

2. For each $i \leq k$, $\text{checkConflict}(U, o_i)$

3. Based on 2, split in two lists:
   - Non-conflict: $p_1 \ldots p_m$
   - Conflict: $q_1 \ldots q_n$

4. For each $i \leq m$, $\text{getScore}(U, p_i)$
For PC member $U$, during Discussion, add new action $\text{listPapersSorted}(U)$ that lists all papers sorted by score.

1. $\text{listPapers}(U)$: output $o_1 \ldots o_k$

2. For each $i \leq k$, $\text{checkConflict}(U, o_i)$

3. Based on 2, split in two lists:
   - Non-conflict: $p_1 \ldots p_m$
   - Conflict: $q_1 \ldots q_n$

4. For each $i \leq m$, $\text{getScore}(U, p_i)$

5. Based on 4, sort $p_1 \ldots p_m$, obtaining $r_1 \ldots r_m$
Example for CoCon++

For PC member $U$, during Discussion, add new action $\text{listPapersSorted}(U)$ that lists all papers sorted by score.

1. $\text{listPapers}(U)$: output $o_1 \ldots o_k$

2. For each $i \leq k$, $\text{checkConflict}(U, o_i)$

3. Based on 2, split in two lists:
   - Non-conflict: $p_1 \ldots p_m$
   - Conflict: $q_1 \ldots q_n$

4. For each $i \leq m$, $\text{getScore}(U, p_i)$

5. Based on 4, sort $p_1 \ldots p_m$, obtaining $r_1 \ldots r_m$

Final output of $\text{listPapersSorted}(U)$: $(r_1 \ldots r_m, q_1 \ldots q_n)$
Example for CoCon++

For PC member $U$, during Discussion, add new action $\text{listPapersSorted}(U)$ that lists all papers sorted by score.

1. $\text{listPapers}(U)$: output $o_1 \ldots o_k$

2. For each $i \leq k$, $\text{checkConflict}(U, o_i)$

3. Based on 2, split in two lists:
   - Non-conflict: $p_1 \ldots p_m$
   - Conflict: $q_1 \ldots q_n$

4. For each $i \leq m$, $\text{getScore}(U, p_i)$

5. Based on 4, sort $p_1 \ldots p_m$, obtaining $r_1 \ldots r_m$

Final output of $\text{listPapersSorted}(U)$: $(r_1 \ldots r_m, q_1 \ldots q_n)$

Summary: old action invocation
Example for CoCon++

For PC member $U$, during Discussion, add new action $\text{listPapersSorted}(U)$ that lists all papers sorted by score.

1. $\text{listPapers}(U)$: output $o_1 \ldots o_k$

2. For each $i \leq k$, $\text{checkConflict}(U, o_i)$

3. Based on 2, split in two lists:
   - Non-conflict: $p_1 \ldots p_m$
   - Conflict: $q_1 \ldots q_n$

4. For each $i \leq m$, $\text{getScore}(U, p_i)$

5. Based on 4, sort $p_1 \ldots p_m$, obtaining $r_1 \ldots r_m$

Final output of $\text{listPapersSorted}(U)$: $(r_1 \ldots r_m, q_1 \ldots q_n)$

Summary: old action invocation + some output filtering
Given a confidentiality-verified system

Extend the system (either with “convenience features” or with information sent on additional channels) as follows:

Each newly added action:

First invokes sequences of old actions, each chosen based on the output of previous actions only

Then synthesizes output from these old actions’ outputs

In particular, does not “peek” directly into the state!

But is this enough to preserve confidentiality?
Given a confidentiality-verified system

Extend the system (either with “convenience features” or with information sent on additional channels) as follows:

Each newly added action:

First invokes sequences of old actions, each chosen based on the output of previous actions only

Then synthesizes output from these old actions’ outputs

In particular, does not “peek” directly into the state!

But is this enough to preserve confidentiality?

In general, NO.
System nondeterministically produces secret bit $i \in \{0, 1\}$. Each user $U$ can take any of the two actions:
System nondeterministically produces secret bit $i \in \{0, 1\}$. Each user $U$ can take any of the two actions:

- **intention($U$)**: no output, but system remembers $U$ and forgets previously remembered user
- **confirmation($U$)**: if remembered user is $U$, return $i$ else return random bit $j$
System nondeterministically produces secret bit $i \in \{0, 1\}$. Each user $U$ can take any of the two actions:

- **intention($U$):** no output, but system remembers $U$ and forgets previously remembered user
- **confirmation($U$):**
  - if remembered user is $U$, return $i$
  - else return random bit $j$
System nondeterministically produces secret bit $i \in \{0, 1\}$. Each user $U$ can take any of the two actions:

- **intention($U$):** no output, but system remembers $U$ and forgets previously remembered user
- **confirmation($U$):**
  - if remembered user is $U$, return $i$
  - else return random bit $j$

Users cannot learn the secret $i$
Counterexample

System nondeterministically produces secret bit $i \in \{0, 1\}$. Each user $U$ can take any of the two actions:

- **intention($U$):** no output, but system remembers $U$ and forgets previously remembered user
- **confirmation($U$):**
  - if remembered user is $U$, return $i$
  - else return random bit $j$

Users cannot learn the secret $i$

Extension with “convenience” feature: user $U$ allowed to chain intention($U$) and confirmation($U$) in one single atomic step.
System nondeterministically produces secret bit $i \in \{0, 1\}$. Each user $U$ can take any of the two actions:

- **intention($U$)**: no output, but system remembers $U$ and forgets previously remembered user
- **confirmation($U$)**:
  
  - if remembered user is $U$, return $i$
  - else return random bit $j$

Users cannot learn the secret $i$

Extension with “convenience” feature: user $U$ allowed to chain $\text{intention}(U)$ and $\text{confirmation}(U)$ in one single atomic step. Then secret $i$ is revealed.
Given a confidentiality-verified system

Extend the system (either with “convenience features” or with information sent on additional channels) as follows:

Each newly added action:

- First invokes sequences of old actions, each chosen based on the output of previous actions only
- Then synthesizes output from these old actions’ outputs

But is this enough to preserve confidentiality?

In general, NO.
Given a confidentiality-verified system

Extend the system (either with “convenience features” or with information sent on additional channels) as follows:

Each newly added action:

First invokes sequences of old actions, each chosen based on the output of previous actions only

Then synthesizes output from these old actions’ outputs

But is this enough to preserve confidentiality?

In general, NO. But for CoCon++ done carefully, probably YES.
Given a confidentiality-verified system

Extend the system (either with “convenience features” or with information sent on additional channels) as follows:

Each newly added action:

- First invokes sequences of old actions, each chosen based on the output of previous actions only
- Then synthesizes output from these old actions’ outputs

But is this enough to preserve confidentiality?

In general, NO. But for CoCon++ done carefully, probably YES.

Why?
Given a confidentiality-verified system

Extend the system (either with “convenience features” or with information sent on additional channels) as follows:

Each newly added action:

- First invokes sequences of old actions, each chosen based on the output of previous actions only
- Then synthesizes output from these old actions’ outputs

But is this enough to preserve confidentiality?

In general, NO. But for CoCon++ done carefully, probably YES.

Why? And can we give the answer for a general class of systems?
Borrowing BD Security in an Ideal World

Adjustment property: \( \forall \, \text{tr}_1, \, \text{ol}_1' \). \( \text{O}_{\text{tr}_1} = \rho \text{ol}_1' \rightarrow \exists \, \text{tr}_{11} \in PB_{\text{ol}_1'} \). \( \text{O}_{\text{tr}_{11}} = \text{O}_{\text{tr}_1} \land S_{\text{tr}_{11}} = S_{\text{tr}_1} \)
Borrowing BD Security in an Ideal World

\[
\begin{align*}
\text{Obs}'^* & \xrightarrow{\rho} \text{Obs}^* \\
\text{SystemTrace}' & \xrightarrow{\pi} \text{SystemTrace} \\
\text{Secret}^* & \xrightarrow{\text{tr}'} \\
\end{align*}
\]

Adjustment property:
\[
\forall \text{tr}'_1, \text{ol}'. \quad \text{O} \text{tr}'_1 = \rho \text{ol}' \Rightarrow \exists \text{tr}'_11 \in \text{PB} \text{ol}'. \quad \text{O} \text{tr}'_11 = \text{O} \text{tr}'_1 \land \text{S} \text{tr}'_11 = \text{S} \text{tr}'_1
\]
Borrowing BD Security in an Ideal World

\[
\begin{array}{c}
\text{Obs'}^* \xrightarrow{\rho} \text{Obs}^* \\
\downarrow O' \quad \uparrow O \\
\text{SystemTrace'} \xrightarrow{\pi} \text{SystemTrace} \\
\downarrow tr' \\
\end{array}
\]

\[
\begin{array}{c}
\text{SystemTrace'} \xrightarrow{\pi} \text{SystemTrace} \\
\downarrow S' \\
\downarrow S \\
\downarrow \text{sl B sl}_1 \text{Secret}^* \\
\end{array}
\]

Adjustment property:

\[
\forall tr_1, ol'. O tr_1 = \rho ol' \Rightarrow \exists tr_11 \in PB ol'. O tr_11 = O tr_1 \land S tr_11 = S tr_1
\]
Borrowing BD Security in an Ideal World

Adjustment property: \( \forall tr_1, ol'. O \quad tr_1 = \rho ol' \rightarrow \exists tr_{11} \in PB \quad ol'. O \quad tr_{11} = O \quad tr_1 \land S \quad tr_{11} = S \quad tr_1 \)
**Adjustment property:**
\[ \forall (tr_1, ol') \in \text{Obs'} \times \text{O} \]
\[ \text{Obs'} \xrightarrow{\rho} \text{Obs}^* \]
\[ \text{SystemTrace'} \xrightarrow{\pi} \text{SystemTrace} \]
\[ \text{Secret}^* \xrightarrow{sl B \ sl_1} \]

\[ \text{Obs'} \xrightarrow{ol'} \text{O'} \]
\[ \text{SystemTrace'} \xrightarrow{tr'} \text{SystemTrace} \xrightarrow{tr} \]
\[ \text{SystemTrace} \xrightarrow{S'} \text{Secret}^* \]

\[ \text{Obs}^* \xrightarrow{O} \]

\[ \text{Obs'} \xrightarrow{\rho} \text{Obs}^* \]

\[ \text{SystemTrace'} \xrightarrow{\pi} \text{SystemTrace} \]

\[ \text{Secret}^* \xrightarrow{sl B \ sl_1} \]
Borrowing BD Security in an Ideal World

\[
\begin{array}{c}
\text{Obs'}^* \quad \rho \quad \text{Obs}^* \\
\downarrow \quad \uparrow \quad \uparrow \\
\text{O'} \quad \text{O} \\
\downarrow \quad \downarrow \\
\text{SystemTrace}' \quad \pi \quad \text{SystemTrace} \\
\downarrow \quad \downarrow \\
\text{tr}' \quad \text{tr} \quad \text{tr}_1 \\
\downarrow \\
\text{s}' \\
\downarrow \\
\text{sl} \ B \ sl_1 \text{Secret}^*
\end{array}
\]

Adjustment property: 
\[
\forall \text{tr}_1 \in \text{PB} \text{ol}' \quad \exists \text{tr}_{11} \in \text{PB} \text{ol}' \quad \text{O tr}_1 = \rho \text{ol}' \quad \text{O tr}_{11} = \text{O tr}_1 \land \text{S tr}_{11} = \text{S tr}_1
\]
Borrowing BD Security in an Ideal World

\[
\begin{align*}
\text{Obs'\,}^* & \xrightarrow{\rho} \text{Obs}\,^* \\
\text{ol'} & \xrightarrow{\pi} \text{SystemTrace'} \\
\text{O'} & \xrightarrow{\pi} \text{SystemTrace} \\
\text{tr'} & \xrightarrow{\pi} \text{SystemTrace} \\
\text{sl} & \xrightarrow{\pi} \text{Secret}\,^* \\
\text{sl_1} & \xrightarrow{\pi} \text{Secret}\,^*
\end{align*}
\]

Adjustment property:
\[
\forall \text{tr}\,^1, \text{ol'} \\Rightarrow \exists \text{tr}\,^1 \in \text{PB} \quad \text{ol} = \text{ol}_1
\]
Borrowing BD Security in an Ideal World

\[ \text{Obs'}^* \xrightarrow{\rho} \text{Obs}^* \]

\[ \text{SystemTrace'} \xrightarrow{\pi} \text{SystemTrace} \]

\[ \text{gl} \]

\[ \forall \text{tr}_1, \text{ol'} \rightarrow \exists \text{tr}_{11} \in \text{PB} \text{ol'}^* \ x \text{tr}_{11} = \text{tr}_1 \land S_{\text{tr}_{11}} = S_{\text{tr}_1} \]
Borrowing BD Security in an Ideal World

\[ \text{Obs'}^{*} \xrightarrow{\rho} \text{Obs}^{*} \]

weak pullback

\[ \text{SystemTrace'} \xrightarrow{\pi} \text{SystemTrace} \]

\[ \text{ol'} \]

\[ O' \]

\[ \text{SystemTrace'} \]

\[ \text{tr'} \]

\[ S' \]

\[ sl \ B \ sl_1 \text{Secret'}^{*} \]

Adjustment property:

\[ \forall \text{tr}_1, \text{ol'} \in \text{Obs'}^{*}, \text{O tr}_1 = \rho \text{ol'} \Rightarrow \exists \text{tr}_1, \text{ol} = \text{ol}_1 \in \text{PB} \text{ol'} \Rightarrow \text{O tr}_1 = \text{O tr}_1 \wedge S \text{tr}_1 = S \text{tr}_1 \]
Borrowing BD Security in an Ideal World

\[ \text{Obs'} \xrightarrow{\rho} \text{Obs}^* \]

\[ \text{O'} \xrightarrow{\text{weak pullback}} O \]

\[ \text{SystemTrace'} \xrightarrow{\pi} \text{SystemTrace} \]

\[ \text{tr'} \xrightarrow{\text{tr'}_1} \]

\[ \text{s'} \xrightarrow{s} \]

\[ \text{sl B sl}_1 \rightarrow \text{Secret}^* \]

Adjustment property:
\[ \forall \text{tr}_1, \text{ol'} \exists \text{tr'}_1 \in \text{PB ol'} \rightarrow \text{ol = ol}_1 \]
Borrowing BD Security in an Ideal World

\[ \text{Obs'}^* \xrightarrow{\rho} \text{Obs}^* \]

\[ \text{SystemTrace'} \xrightarrow{\pi} \text{SystemTrace} \]

\[ \text{Obs'}^* \xrightarrow{\rho} \text{Obs}^* \]

\[ \text{SystemTrace'} \xrightarrow{\pi} \text{SystemTrace} \]

\[ \text{Secret}^* \]

Adjustment property:

\[ \forall \text{tr}_1, \text{ol'} \in \text{PB}, \text{O} \text{tr}_1 = \rho \text{ol'} \rightarrow \exists \text{tr}_11 \in \text{PB}, \text{O} \text{tr}_11 = \text{O} \text{tr}_1 \wedge S \text{tr}_11 = S \text{tr}_1 \]
Borrowing BD Security in a Real World

\[ \text{Obs}^\prime \xrightarrow{\rho} \text{Obs}^\star \]

\[ \text{SystemTrace}^\prime \xrightarrow{\pi} \text{SystemTrace} \]

\[ \text{SlB} \xrightarrow{sl} \text{Secret}^\star \]

\[ \text{Ol} = \text{Ol}_1 \]

Adjustment property:
\[ \forall \text{Tr}_1, \text{Ol}^\prime. \text{Ol} = \text{Ol}_1 \Rightarrow \exists \text{Tr}_1. \text{Ol} = \text{Ol}_1 \wedge S = S \]

weak pullback
Borrowing BD Security in a Real World

Adjustment property: 
\[ \forall tr_1, ol': O_{tr_1} = \rho ol' \rightarrow \exists tr_{11} \in PB ol'. O_{tr_{11}} = O_{tr_1} \land S_{tr_{11}} = S_{tr_1} \]
Borrowing BD Security in a Real World

\[ \text{Obs'}^* \xrightarrow{\rho} \text{Obs}^* \]

\[ \text{O'} \]

\[ \text{SystemTrace'} \xrightarrow{\pi} \text{SystemTrace} \]

\[ \text{tr'} \xrightarrow{\pi} \text{tr}_1 \]

\[ \text{S'} \]

\[ \text{sl B sl}_1 \text{Secret}^* \]

Adjustment property:

\[ \forall \text{tr}_1, \text{ol}' \in \text{PB} \]

\[ \text{O} \xrightarrow{\rho} \text{ol} = \text{ol}_1 \]

\[ \text{O} \xrightarrow{\rho} \text{ol} = \text{ol}_1 \]
Borrowing BD Security in a **Real** World

\[
\begin{align*}
\text{Obs'}^* & \xrightarrow{\rho} \text{Obs}^* \\
\text{SystemTrace'} & \xrightarrow{\pi} \text{SystemTrace} \\
\text{Secret'} & \xrightarrow{\pi} \text{PB}_{\text{ol}'}
\end{align*}
\]
Adjustment property: \( \forall tr_1, ol' \in \text{Obs} \), \( O(tr_1) = \rho(ol') \Rightarrow \exists tr_{11} \in \text{PB}_{ol'}. O(tr_{11}) = O(tr_1) \land S(tr_{11}) = S(tr_1) \)
Borrowing BD Security in a Real World

\[ \text{Obs}* \xrightarrow{\rho} \text{Obs}* \]

\[ \text{SystemTrace}' \xrightarrow{\pi} \text{SystemTrace} \xleftarrow{\text{PB}_{\text{ol}'} } \]

\[ \text{Secret}^* \]

Adjustment property:
\[ \forall \text{tr}_1, \text{ol}'. \quad \text{O} \text{tr}_1 = \rho \text{ol}' \rightarrow \exists \text{tr}_{11} \in \text{PB}_{\text{ol}'} . \quad \text{O} \text{tr}_{11} = \text{O} \text{tr}_1 \land S \text{tr}_{11} = S \text{tr}_1 \]
Borrowing BD Security in a **Real** World

Adjustment property:

∀\(tr \ 1\), \(ol'\).
\(O\.tr 1 = \rho \ ol'\) \(\rightarrow\) ∃\(tr \ 11\) ∈ \(PB_{ol'}\).
\(O\.tr 11 = O\.tr 1 \land S\tr 11 = S\tr 1\).
Adjustment property:
\[ \forall tr_1, ol'. \ O \ tr_1 = \rho \ ol' \rightarrow \exists tr_{11} \in PB_{ol'}. \ O \ tr_{11} = O \ tr_1 \land S \ tr_{11} = S \ tr_1 \]
Borrowing BD Security in a Real World

Adjustment property:
\[ \forall \text{tr}_1, \text{ol}'. \ O \text{tr}_1 = \rho \text{ol}' \rightarrow \exists \text{tr}_{11} \in \text{PB}_{\text{ol}'} . \ O \text{tr}_{11} = O \text{tr}_1 \land S \text{tr}_{11} = S \text{tr}_1 \]
Borrowing BD Security in a **Real World**

**Adjustment property:**
\[ \forall tr_1, ol'. \, \rho (O \, tr_1, \, ol') \rightarrow \exists tr_{11} \in PB_{ol'}. \, O \, tr_{11} = O \, tr_1 \land S \, tr_{11} = S \, tr_1 \]
Adjustment property:
\[ \forall tr_1, ol'. \rho(O \ tr_1, ol') \rightarrow \exists tr_{11} \in PB_{ol'}. \ O \ tr_{11} = O \ tr_1 \land S \ tr_{11} = S \ tr_1 \]
What I Don’t Like About CoCon

It’s not as much fun as other systems...
Email subject: [ESOP15] Accepted paper #68
Email subject: [ESOP15] Accepted paper #68
Email content: The 24th European Symposium on Programming (ESOP15) program committee is sorry to inform you that your paper #68 was rejected, and will not appear in the conference.

Later email: Dear author, SORRY... we meant the paper is accepted! Apologies it was a long day.
Email subject: [ESOP15] Accepted paper #68

Email content: The 24th European Symposium on Programming (ESOP15) program committee is sorry to inform you that your paper #68 was rejected, and will not appear in the conference.

Later email: Dear author, SORRY… we meant the paper is accepted! Apologies it was a long day.
What I Don’t Like About CoCon

It is implemented as a monolithic system
What I Don’t Like About CoCon

It is implemented as a monolithic system

But many interesting workflow systems are distributed
What I Don’t Like About CoCon

It is implemented as a monolithic system

But many interesting workflow systems are distributed

Experimenting with CoCon only we fail to explore BD security compositionality
EPSRC grant: Verification or Web-Based Systems (VOWS)
EPSRC grant: Verification or Web-Based Systems (VOWS)

Confidentiality-Verified Social Media Platform
EPSRC grant: Verification or Web-Based Systems (VOWS)

Confidentiality-Verified Social Media Platform

Implemented similarly to CoCon, but includes primitives for communicating with clones of itself — Clones can form a network
EPSRC grant: Verification or Web-Based Systems (VOWS)

Confidentiality-Verified Social Media Platform

Implemented similarly to CoCon, but includes primitives for communicating with clones of itself — Clones can form a network

Verify each component for confidentiality properties (BD security instance)
EPSRC grant: Verification or Web-Based Systems (VOWS)

Confidentiality-Verified Social Media Platform

Implemented similarly to CoCon, but includes primitives for communicating with clones of itself — Clones can form a network

Verify each component for confidentiality properties (BD security instance)

Compositionality results offer guarantees for the whole network
EPSRC grant: Verification or Web-Based Systems (VOWS)

Confidentiality-Verified Social Media Platform

Implemented similarly to CoCon, but includes primitives for communicating with clones of itself — Clones can form a network

Verify each component for confidentiality properties (BD security instance)

Compositionality results offer guarantees for the whole network

Example:
A non-friend user will learn nothing about the content of friend-restricted documents
EPSRC grant: Verification or Web-Based Systems (VOWS)

Confidentiality-Verified Social Media Platform

Implemented similarly to CoCon, but includes primitives for communicating with clones of itself — Clones can form a network

Verify each component for confidentiality properties (BD security instance)

Compositionality results offer guarantees for the whole network

Example:
A non-friend user will learn nothing about the content of friend-restricted documents even if the document is shared with the client clones
Future Work: End to End Security

- Isabelle Specification
  - code generation
  - Scala Program
    - Scalatra API
    - Web Application

End to end verification:
- Verify thin API layer
- Complement with client-side monitoring

More practical engineering (VOWS):
- Don’t ask the programmer to use Isabelle – let them use a web framework

More scalable verification (VOWS):
- Model-check BD security
Future Work: End to End Security

End to end verification:
Verify thin API layer
Future Work: End to End Security

End to end verification:
- Verify thin API layer
- Complement with client-side monitoring

Diagram:
- Isabelle Specification
- Scala Program
- Scalatra API
- Web Application
- Code generation
Future Work: End to End Security

Isabelle Specification

Scala Program

Scalatra API

Web Application

End to end verification:
- Verify thin API layer
- Complement with client-side monitoring

More practical engineering (VOWS):
- Don’t ask the programmer to use Isabelle – let them use a web framework
Future Work: End to End Security

End to end verification:
- Verify thin API layer
- Complement with client-side monitoring

More practical engineering (VOWS):
- Don’t ask the programmer to use Isabelle – let them use a web framework

More scalable verification (VOWS): Model-check BD security
CoCon is starting to replace EasyChair at mainstream conferences

- TABLEAUX 2015
- ITP 2016
- CAV 2017 chairs are evaluating the system
CoCon is starting to replace EasyChair at mainstream conferences

- TABLEAUX 2015
- ITP 2016
- CAV 2017 chairs are evaluating the system

Develop concepts and tools that will allow **certified** systems like CoCon to entirely replace **uncertified** systems like EasyChair
CoCon is starting to replace EasyChair at mainstream conferences

TABLEAUX 2015
ITP 2016
CAV 2017 chairs are evaluating the system

Develop concepts and tools that will allow **certified** systems like CoCon to entirely replace **uncertified** systems like EasyChair

Towards this goal:

- Improved automation for proof assistants 🔨
- Better abstraction mechanisms
- More robust logical foundations
CoCon

Andrei Popescu

Middlesex University London
Technical University Munich

Joint work with
Peter Lammich, Sergey Grebenshchikov,
Sudeep Kanav and Thomas Bauereiß