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

Technische Universität München

Middlesex, UK

Interactive Theorem Proving

A Verified Conference System

Conference system (think EasyChair, HotCRP) =
web application for submitting and reviewing scientific papers

A Verified Conference System

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.

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.

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.

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 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;`

Implemented and verified CoCon, a conference system with confidentiality guarantees

CoCon

CoCon = Feature-Rich Conference Management System (similar to EasyChair and HotCRP)

https://cocon.in.tum.de

Welcome to CoCon.

CoCon is a conference management system with verified document confidentiality. What makes CoCon special compared to other conference systems is the fact that **quite certainly it does not leak**.

Learn more »

Sign in to CoCon

Email

Password

Forgot your password?

Sign in

New user? Sign up here

Email

Full name

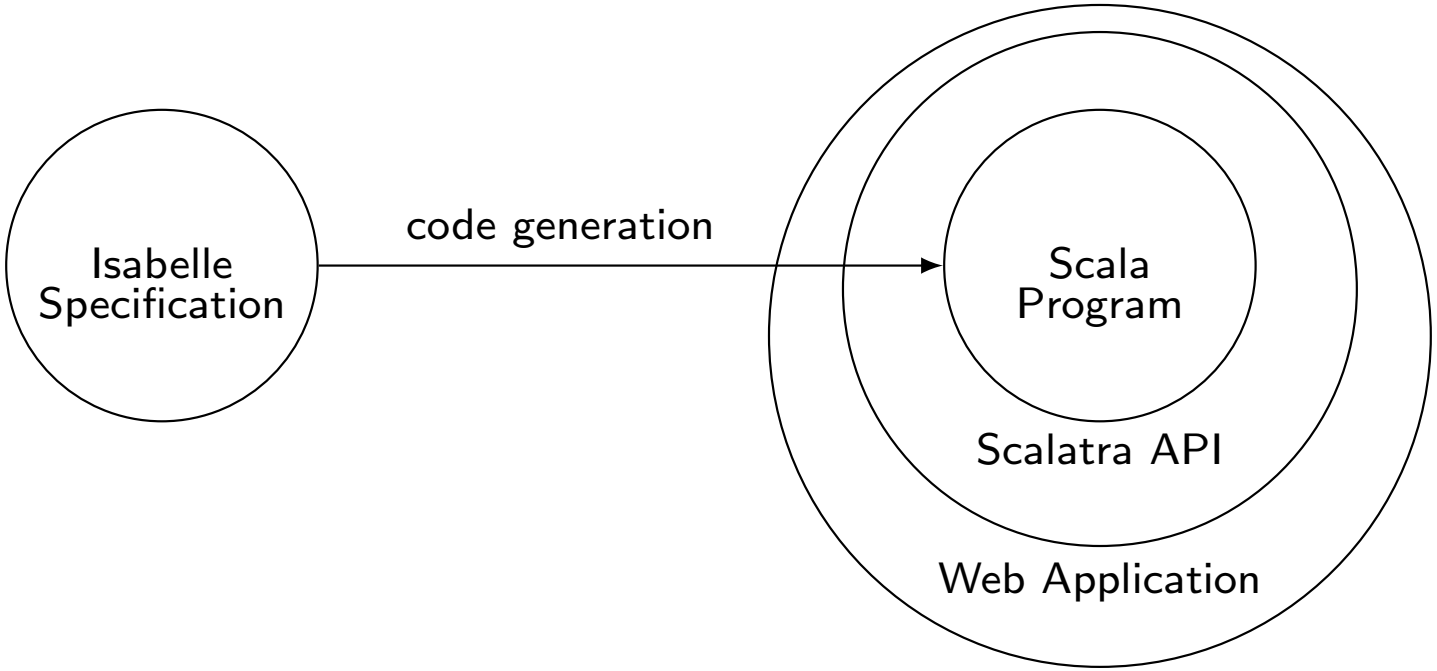
Affiliation

Password

Password (confirmation)

Sign up

CoCon's Architecture



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

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

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

Source Sink Trigger Bound

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

Bounded-Deducibility (BD) Security

$\mathsf{SystemTrace} \subseteq \mathsf{Event}^*$

$\mathsf{SystemTrace}$

Secret^*

Obs^*

Bounded-Deducibility (BD) Security

$\mathsf{SystemTrace} \subseteq \mathsf{Event}^*$

S = produce secrets from selected events

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

Obs^*

Bounded-Deducibility (BD) Security

$\mathsf{SystemTrace} \subseteq \mathsf{Event}^*$

O = produce observations from selected events

$$\begin{tikzcd}
\mathsf{SystemTrace} \arrow[r, "\mathsf{S}"] \arrow[d, "\mathsf{O}"'] & \mathsf{Secret}^* \\
\mathsf{Obs}^* &
\end{tikzcd}$$

Bounded-Deducibility (BD) Security

$\mathsf{SystemTrace} \subseteq \mathsf{Event}^*$

$\mathsf{T} : \mathsf{Event} \rightarrow \mathsf{Bool}$

$$\begin{tikzcd}
\neg\,\mathsf{T} & \mathsf{SystemTrace} \arrow[r, "\mathsf{S}"] \arrow[d, "\mathsf{O}"'] & \mathsf{Secret}^* \\
& \mathsf{Obs}^* &
\end{tikzcd}$$

Bounded-Deducibility (BD) Security

$\mathsf{SystemTrace} \subseteq \mathsf{Event}^*$

$\mathsf{T} : \mathsf{Event} \to \mathsf{Bool} \qquad \mathsf{B}$ relation on Secret^*

$$\begin{tikzcd}
\neg\,\mathsf{T} & \mathsf{SystemTrace} \arrow[r, "\mathsf{S}"] \arrow[d, "\mathsf{O}"'] & \mathsf{Secret}^* & \mathsf{B} \\
& \mathsf{Obs}^* & &
\end{tikzcd}$$

Bounded-Deducibility (BD) Security

$\mathsf{SystemTrace} \subseteq \mathsf{Event}^*$

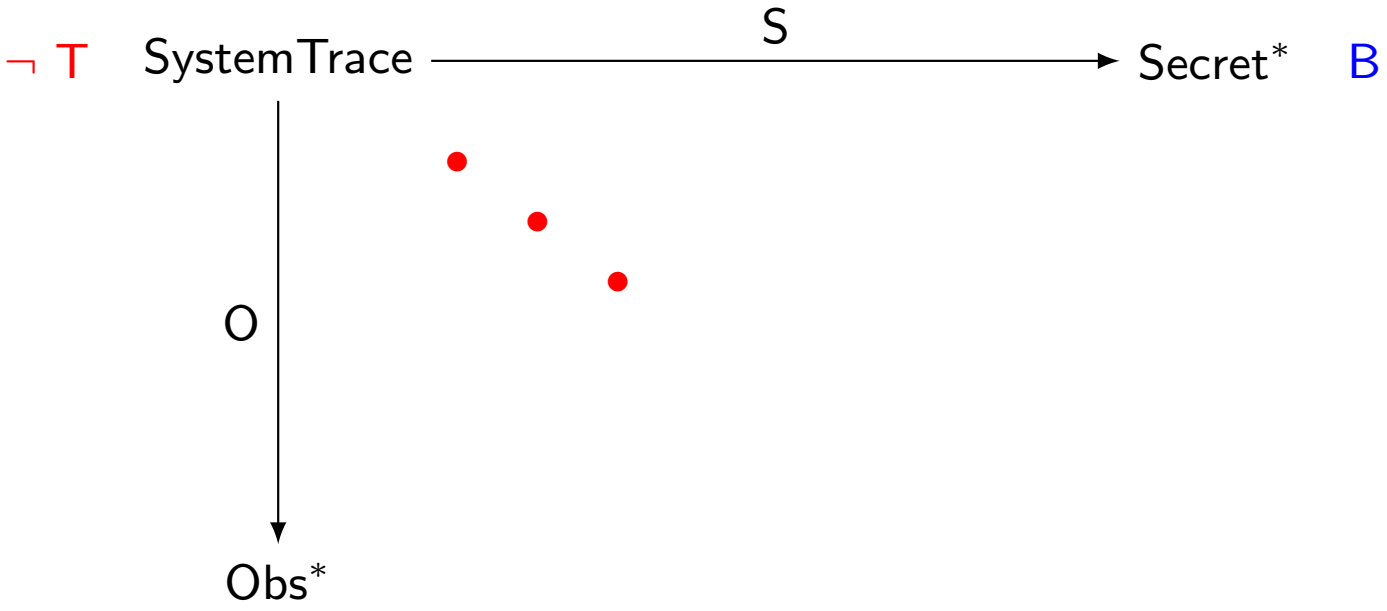
Unless T occurs, O can learn nothing about S beyond B

$$\begin{tikzcd}
\neg\, T \quad \mathsf{SystemTrace} \arrow[r, "S"] \arrow[d, "O"'] & \mathsf{Secret}^* \quad B \\
\mathsf{Obs}^* &
\end{tikzcd}$$

Bounded-Deducibility (BD) Security

$\mathsf{SystemTrace} \subseteq \mathsf{Event}^*$

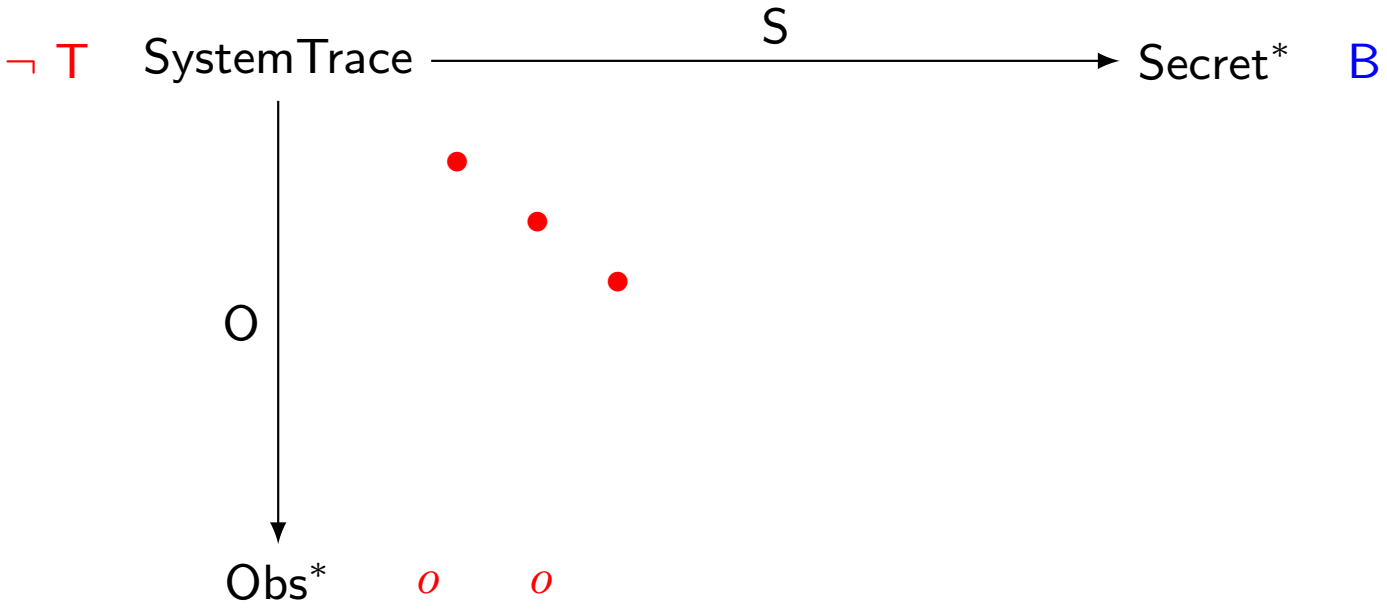
Unless T occurs, O can learn nothing about S beyond B



Bounded-Deducibility (BD) Security

$\mathsf{SystemTrace} \subseteq \mathsf{Event}^*$

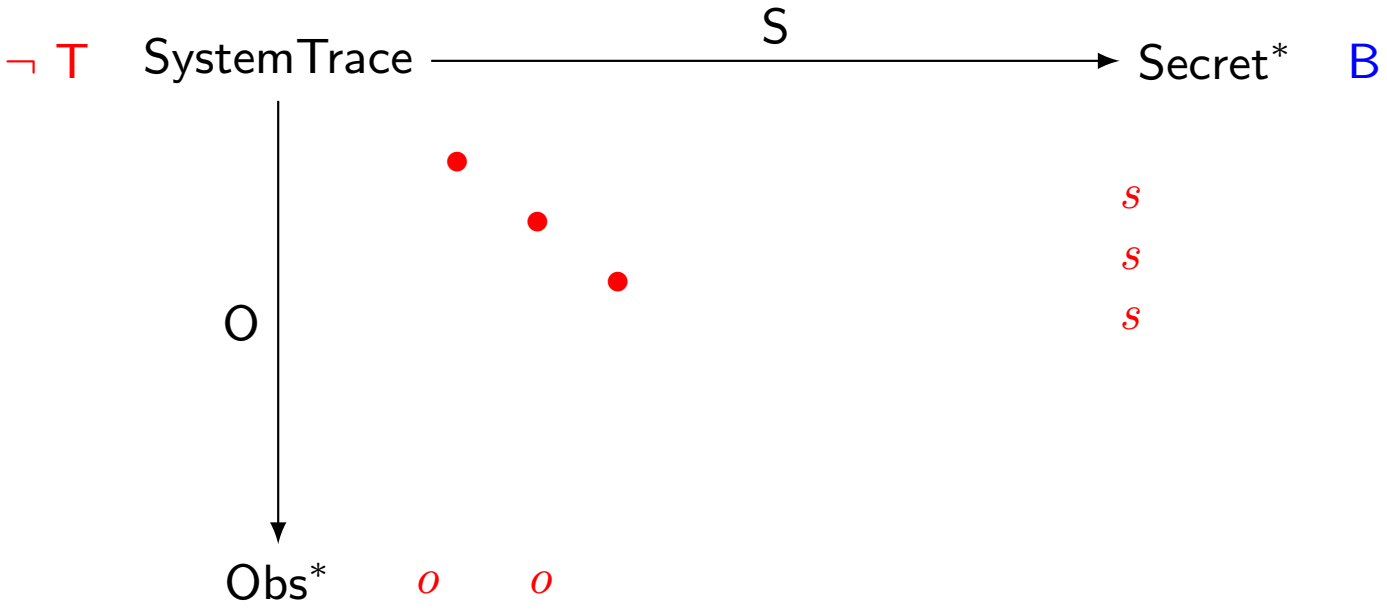
Unless T occurs, O can learn nothing about S beyond B



Bounded-Deducibility (BD) Security

$\mathsf{SystemTrace} \subseteq \mathsf{Event}^*$

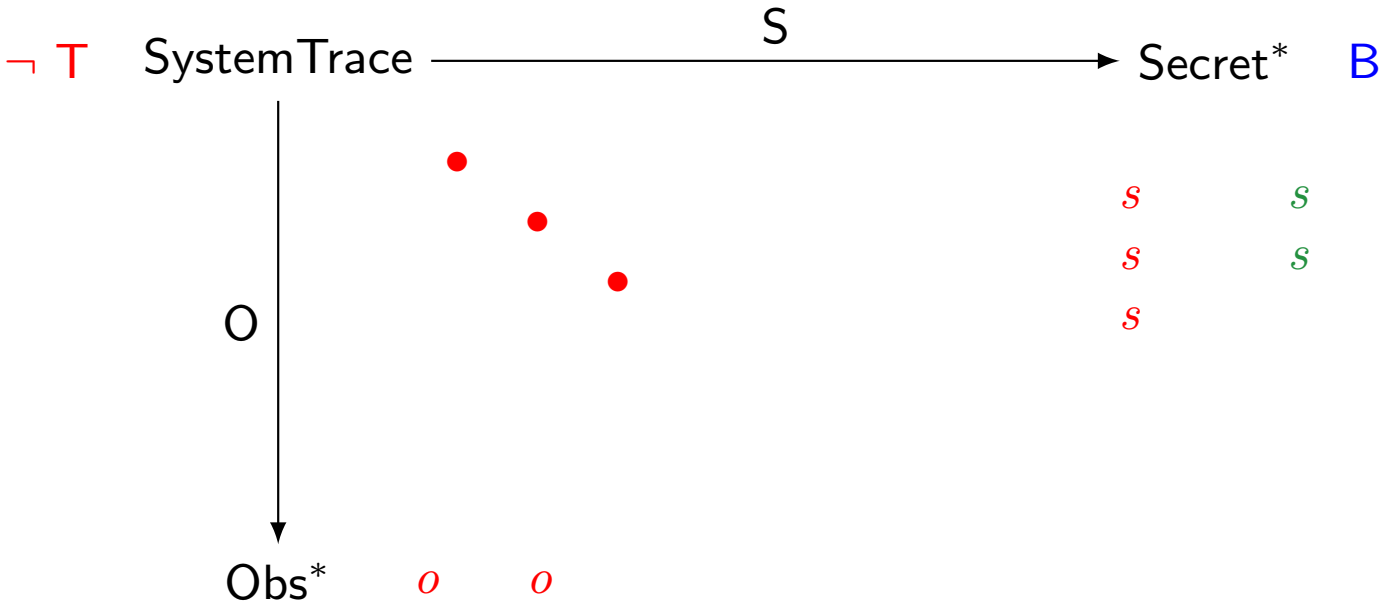
Unless T occurs, O can learn nothing about S beyond B



Bounded-Deducibility (BD) Security

$\mathsf{SystemTrace} \subseteq \mathsf{Event}^*$

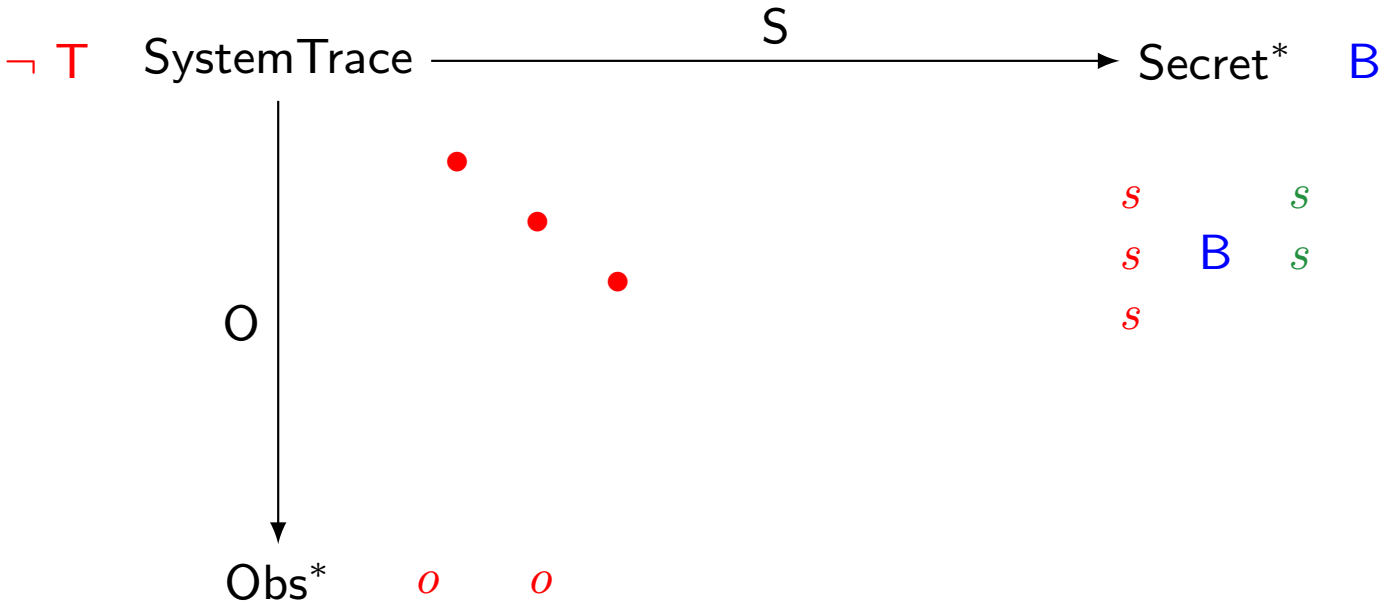
Unless T occurs, O can learn nothing about S beyond B



Bounded-Deducibility (BD) Security

$\mathsf{SystemTrace} \subseteq \mathsf{Event}^*$

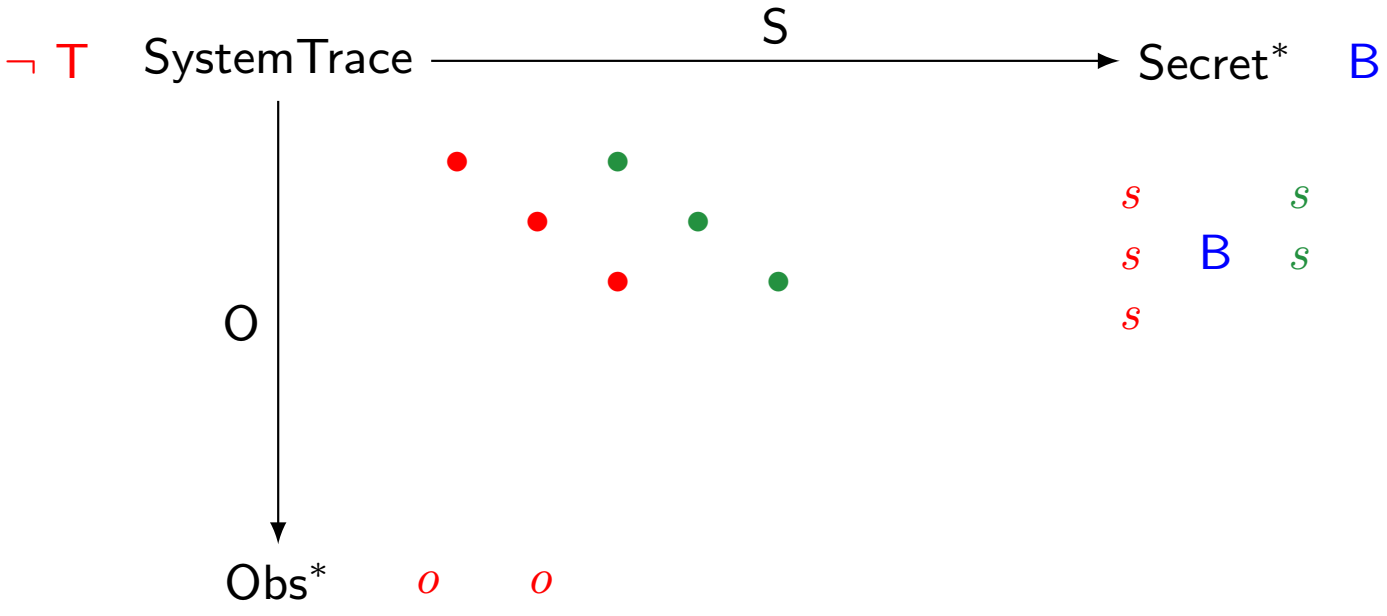
Unless T occurs, O can learn nothing about S beyond B



Bounded-Deducibility (BD) Security

$\mathsf{SystemTrace} \subseteq \mathsf{Event}^*$

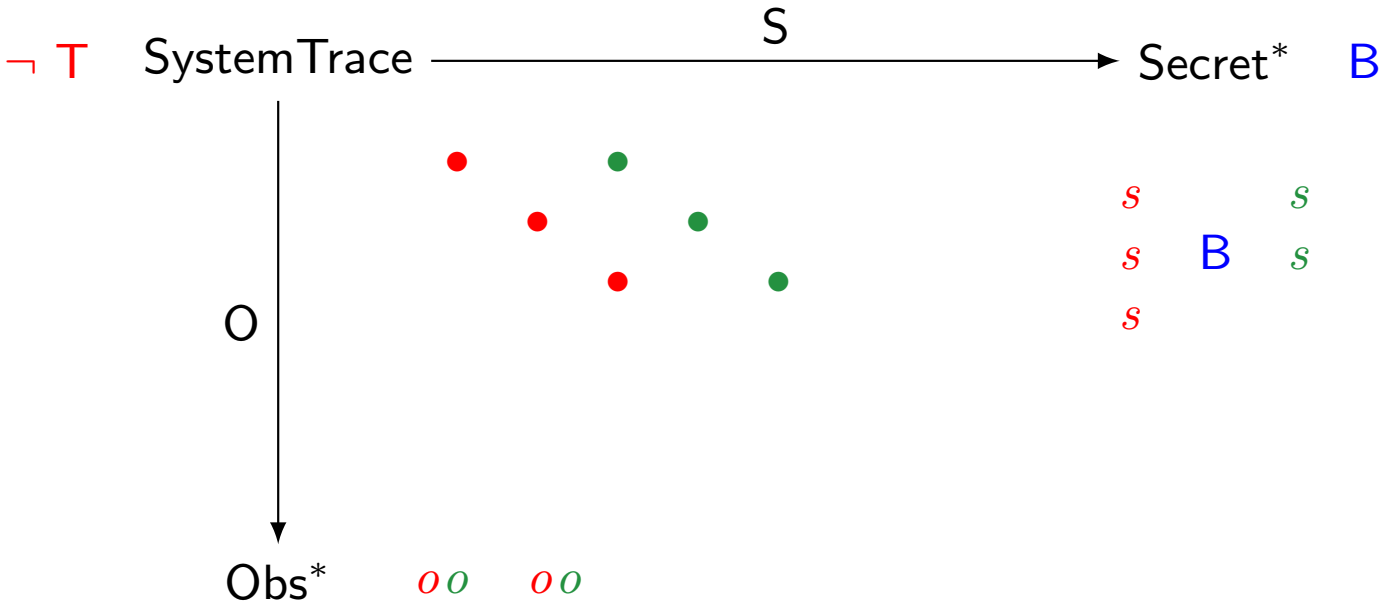
Unless T occurs, O can learn nothing about S beyond B



Bounded-Deducibility (BD) Security

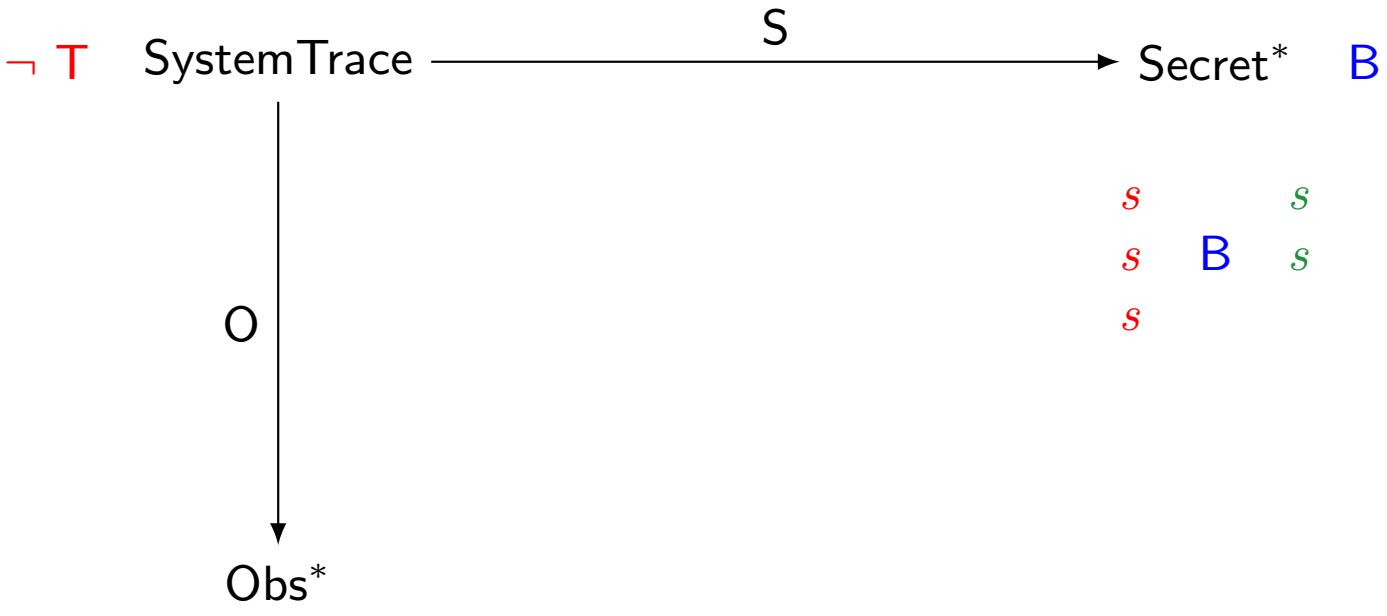
$\mathsf{SystemTrace} \subseteq \mathsf{Event}^*$

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



Bounded-Deducibility (BD) Security

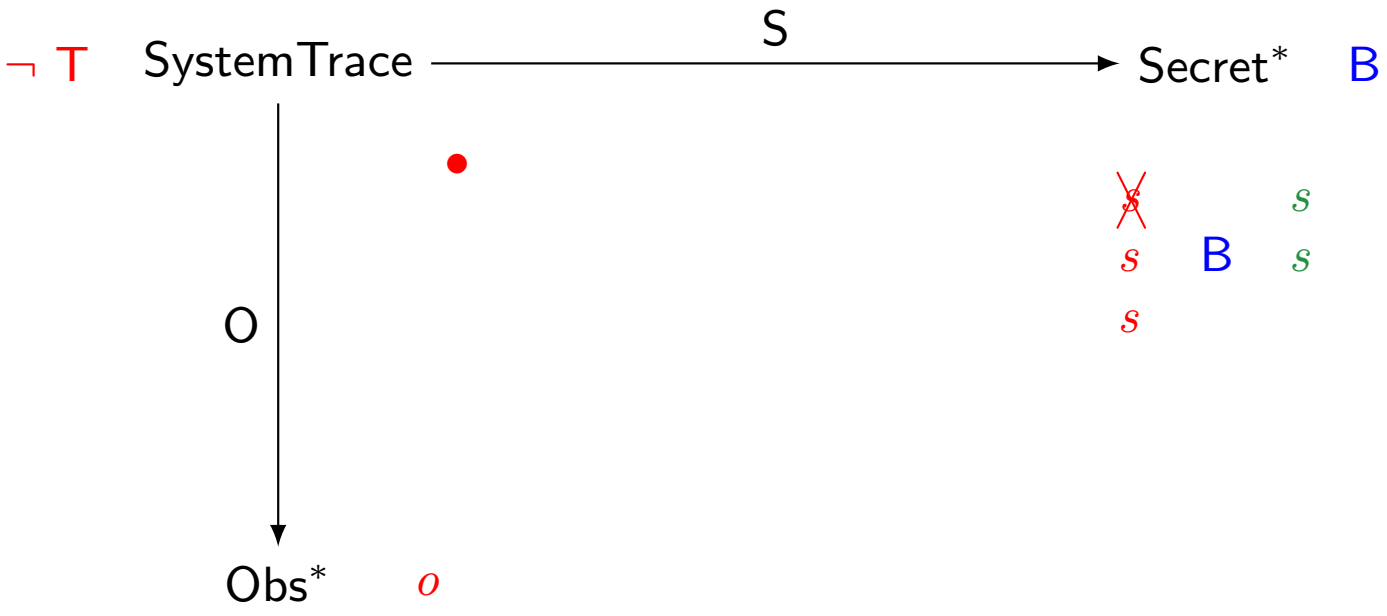
Proof by unwinding



Bounded-Deducibility (BD) Security

Proof by unwinding

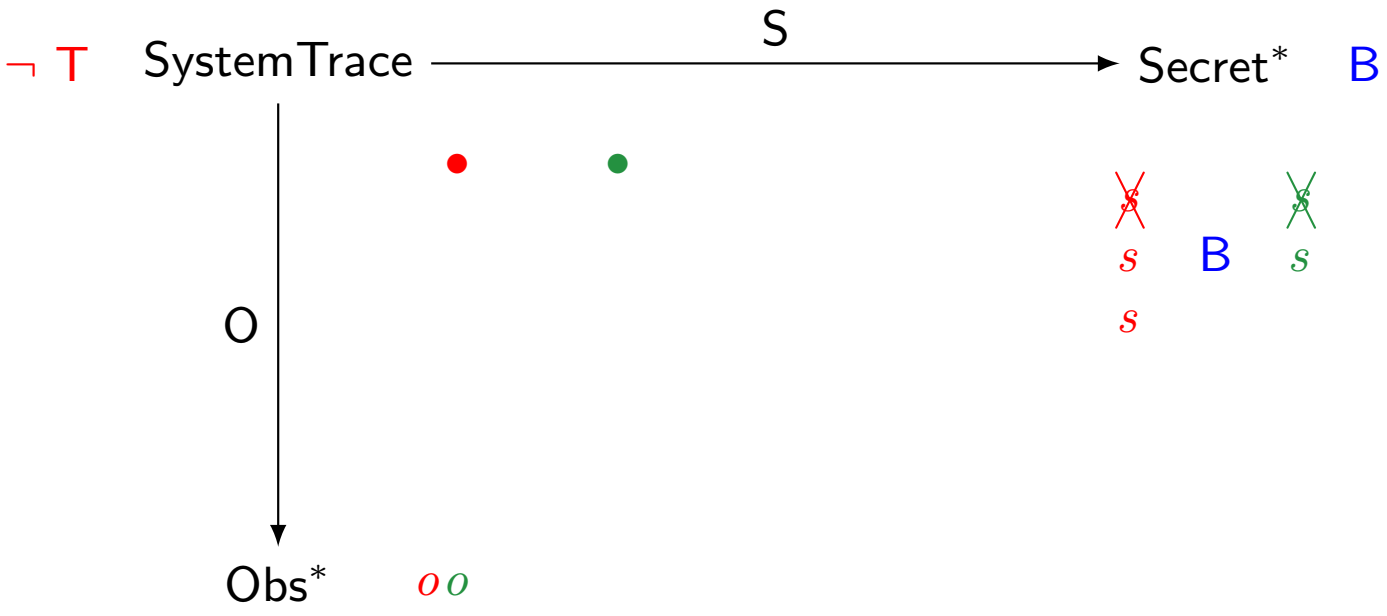
Action



Bounded-Deducibility (BD) Security

Proof by unwinding

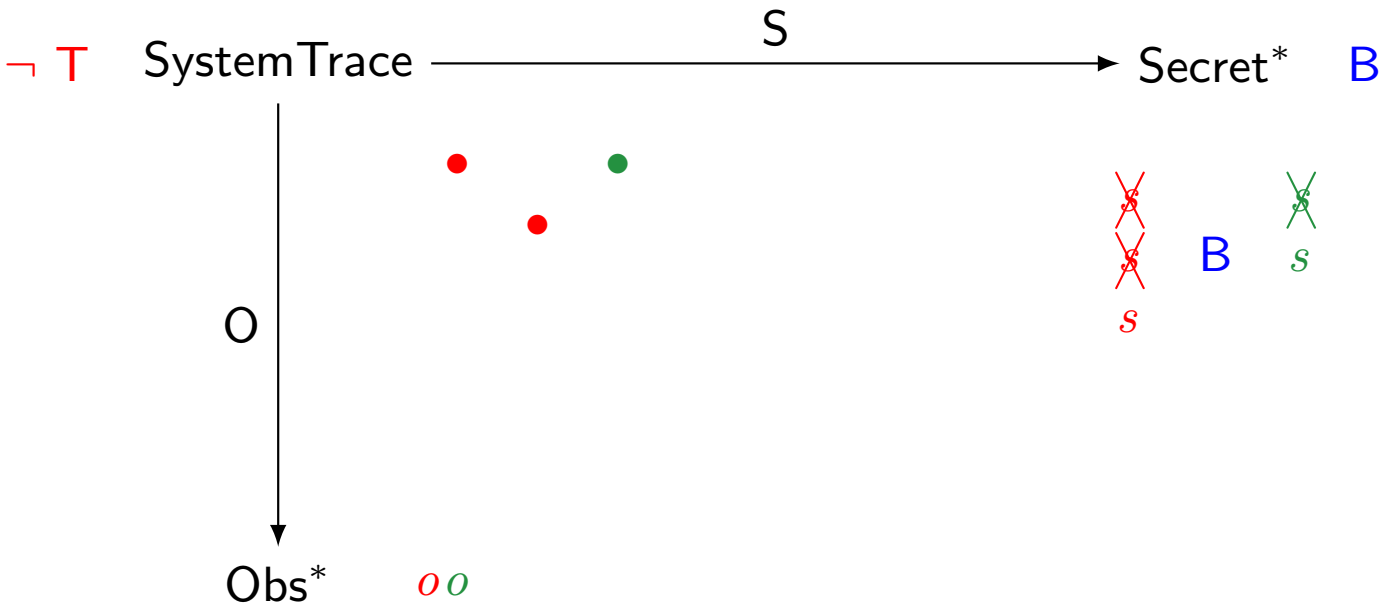
Action / Reaction: Match



Bounded-Deducibility (BD) Security

Proof by unwinding

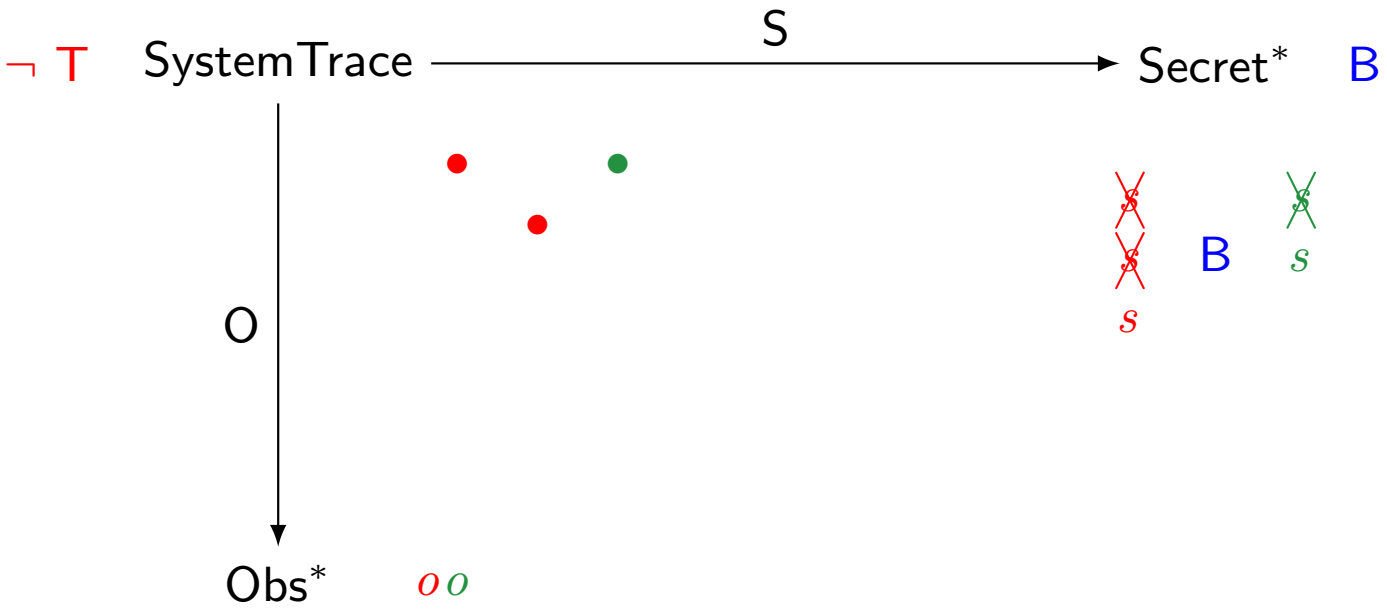
Action



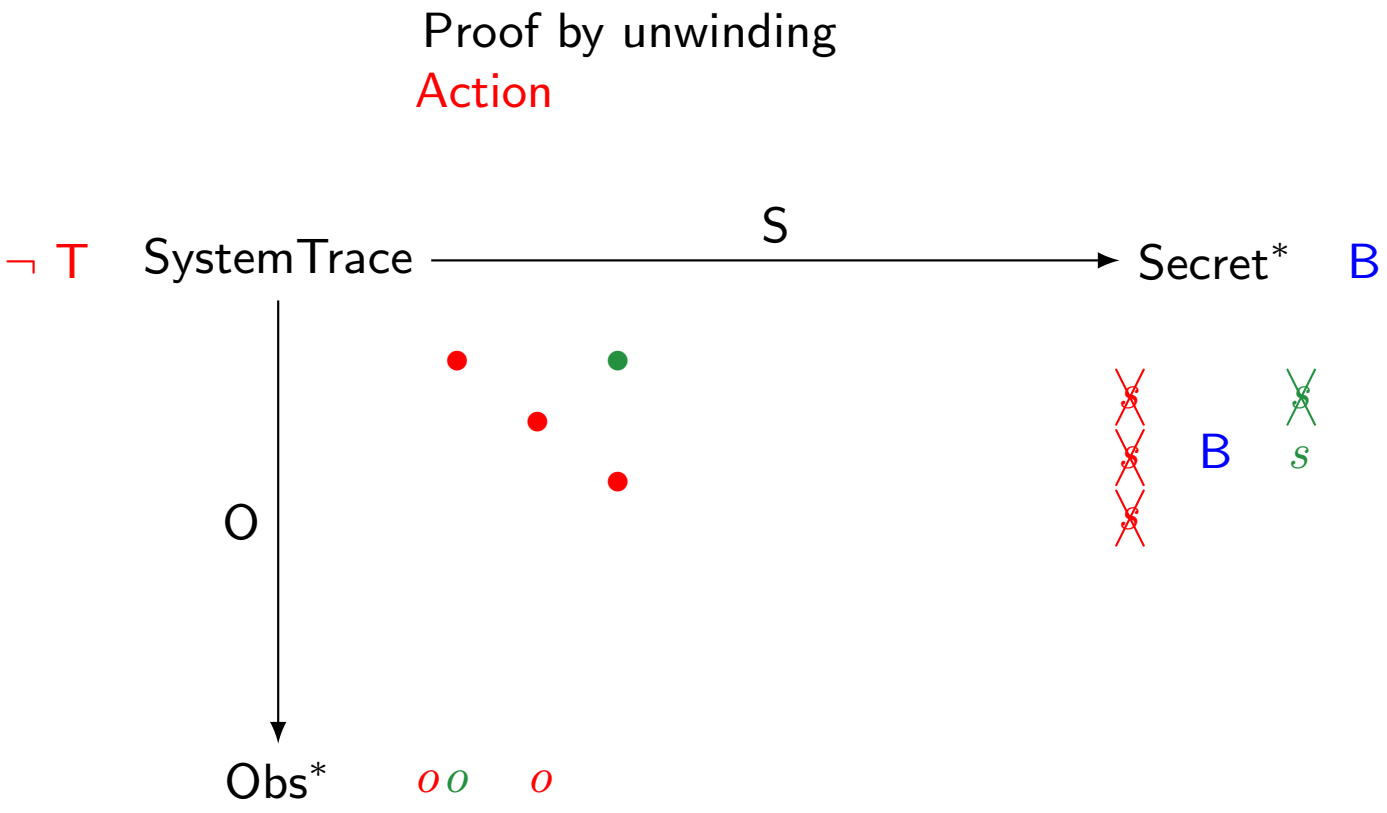
Bounded-Deducibility (BD) Security

Proof by unwinding

Action / Reaction: Ignore



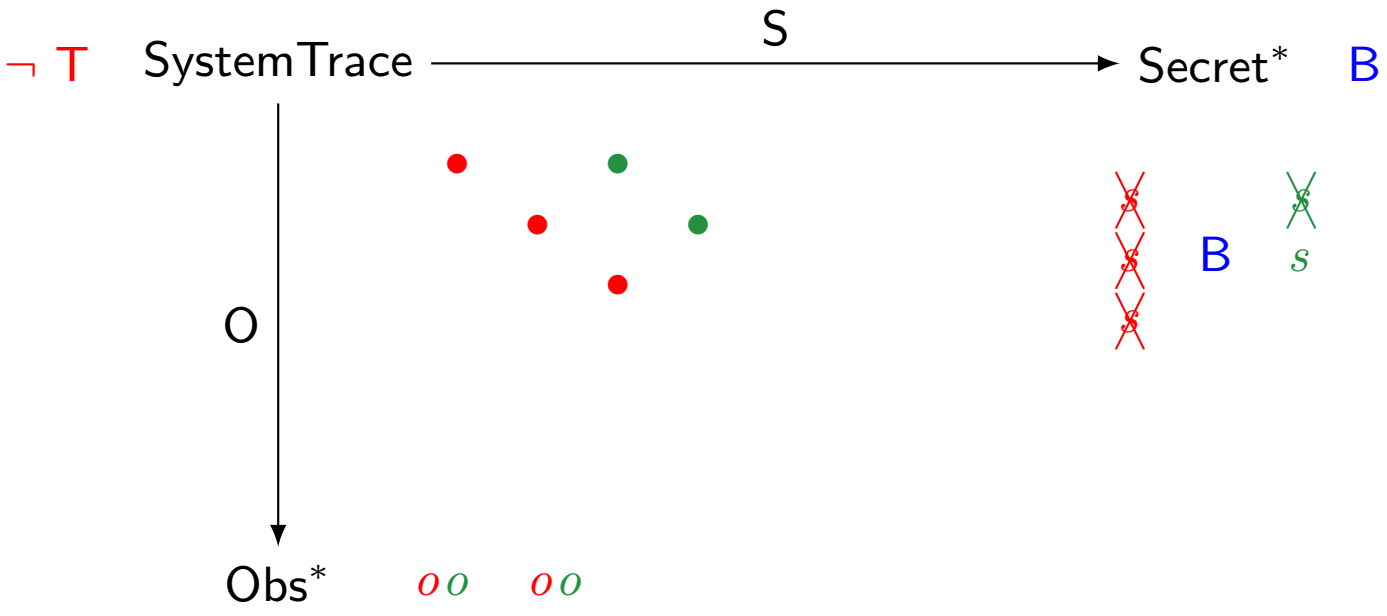
Bounded-Deducibility (BD) Security



Bounded-Deducibility (BD) Security

Proof by unwinding

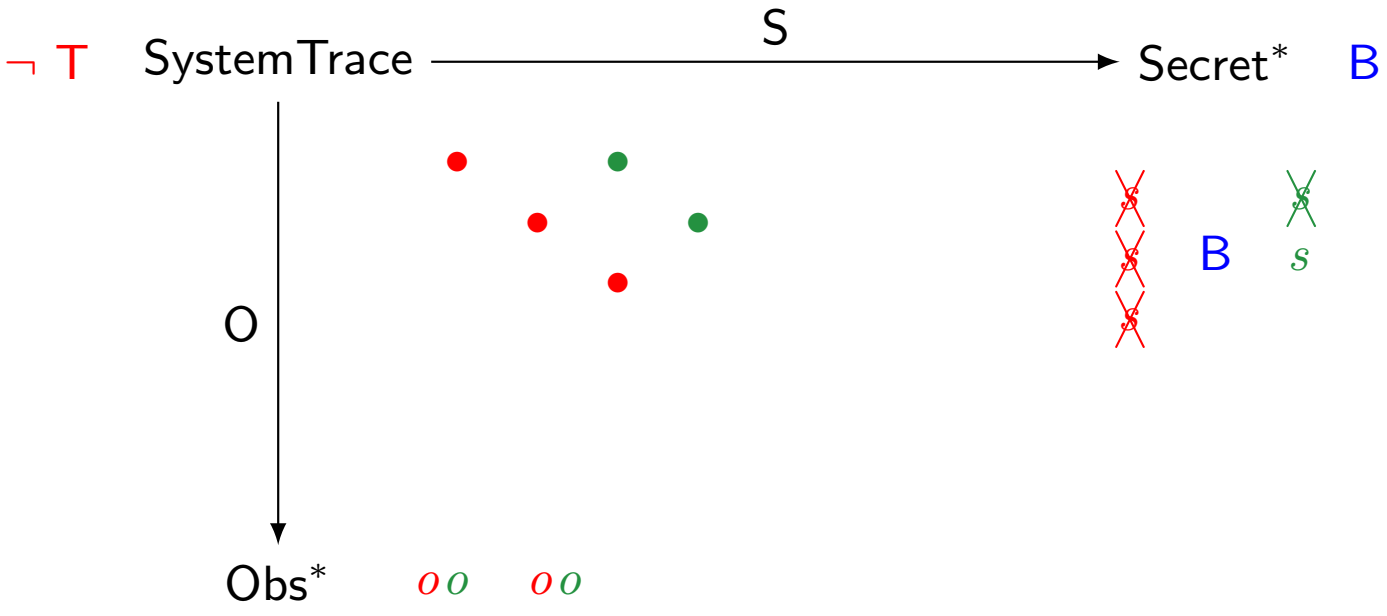
Action / Reaction: Match



Bounded-Deducibility (BD) Security

Proof by unwinding

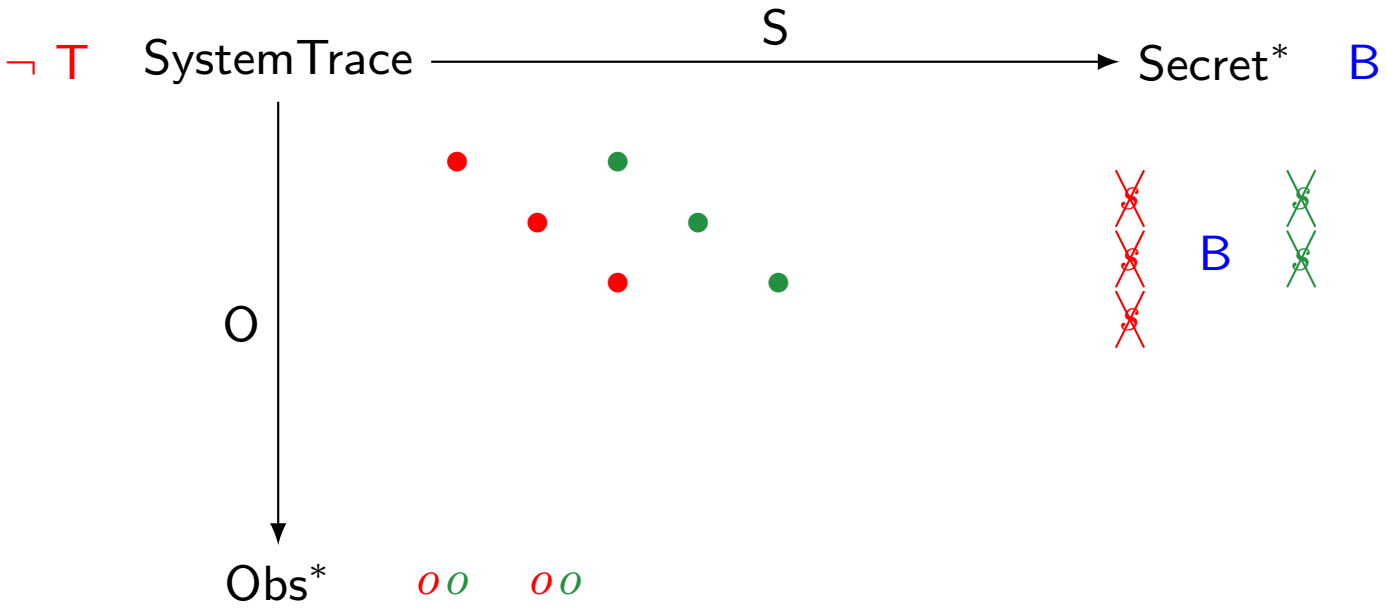
Independent action ...



Bounded-Deducibility (BD) Security

Proof by unwinding

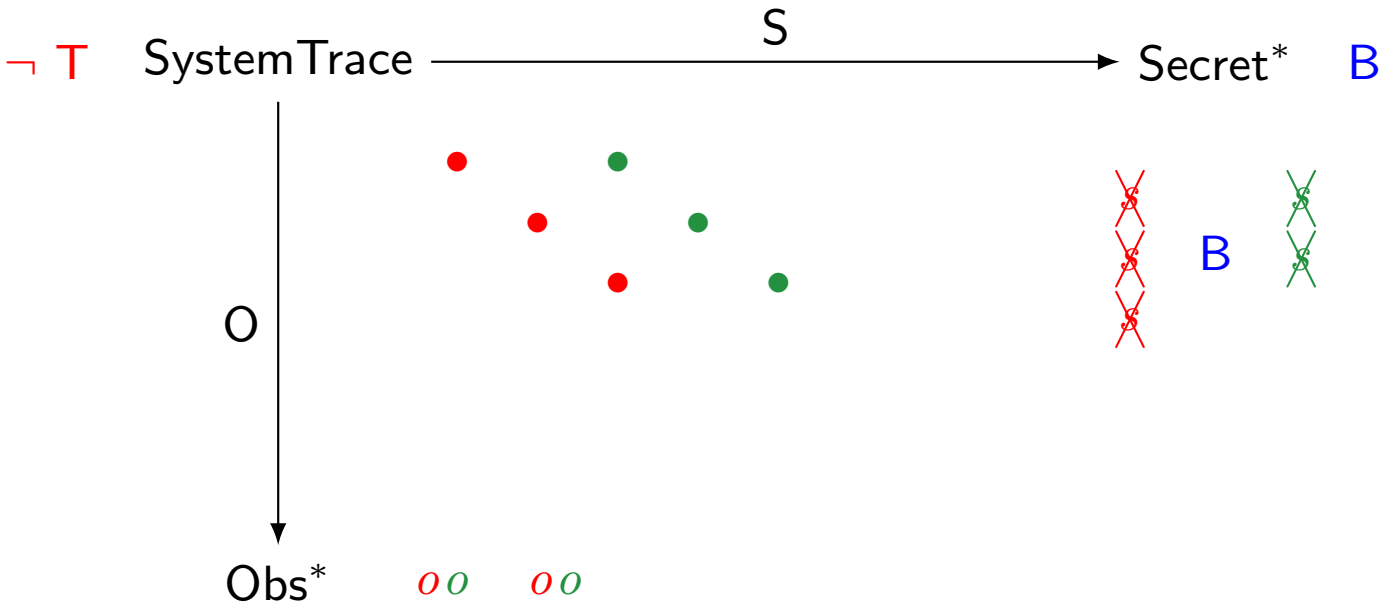
Independent action



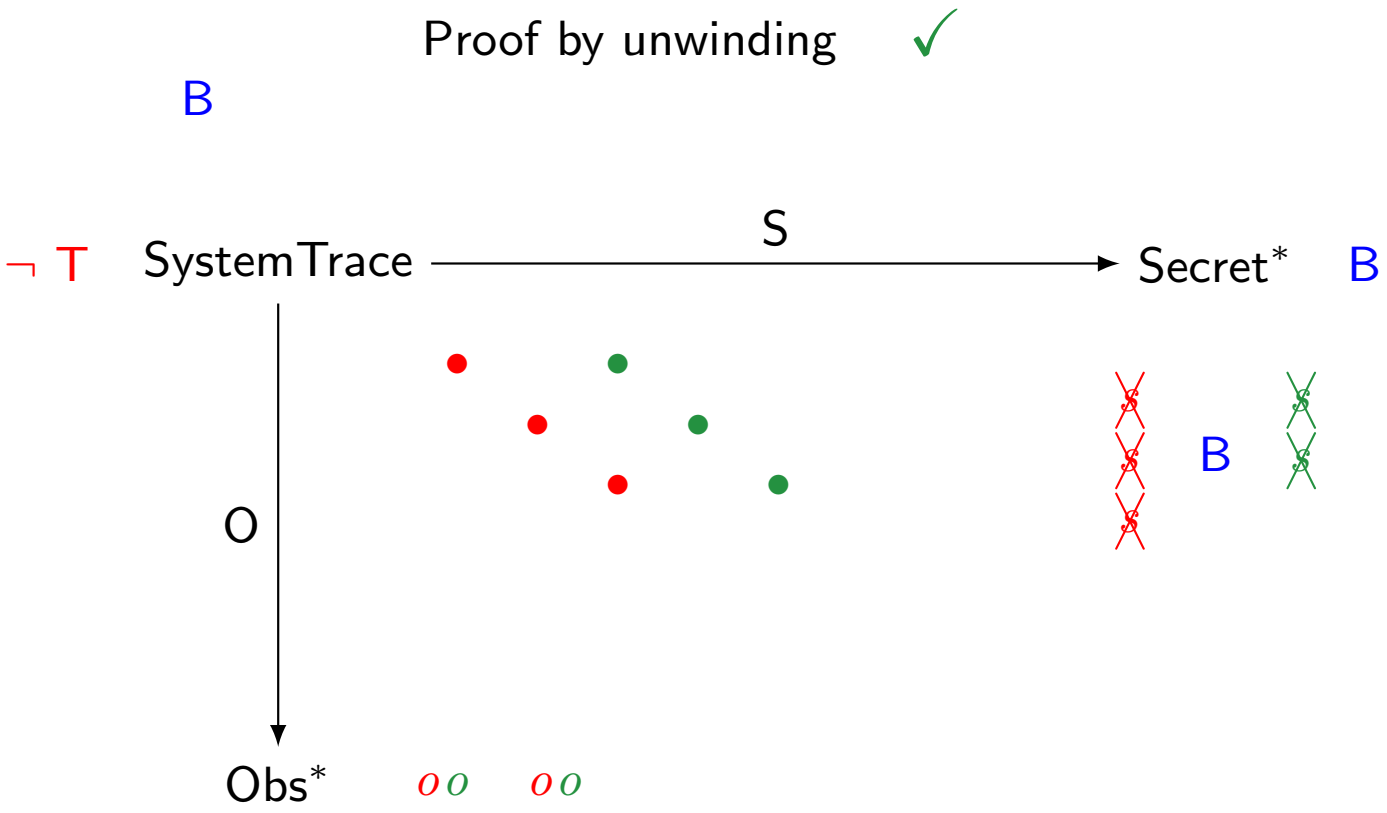
Bounded-Deducibility (BD) Security

Proof by unwinding ✓

Independent action



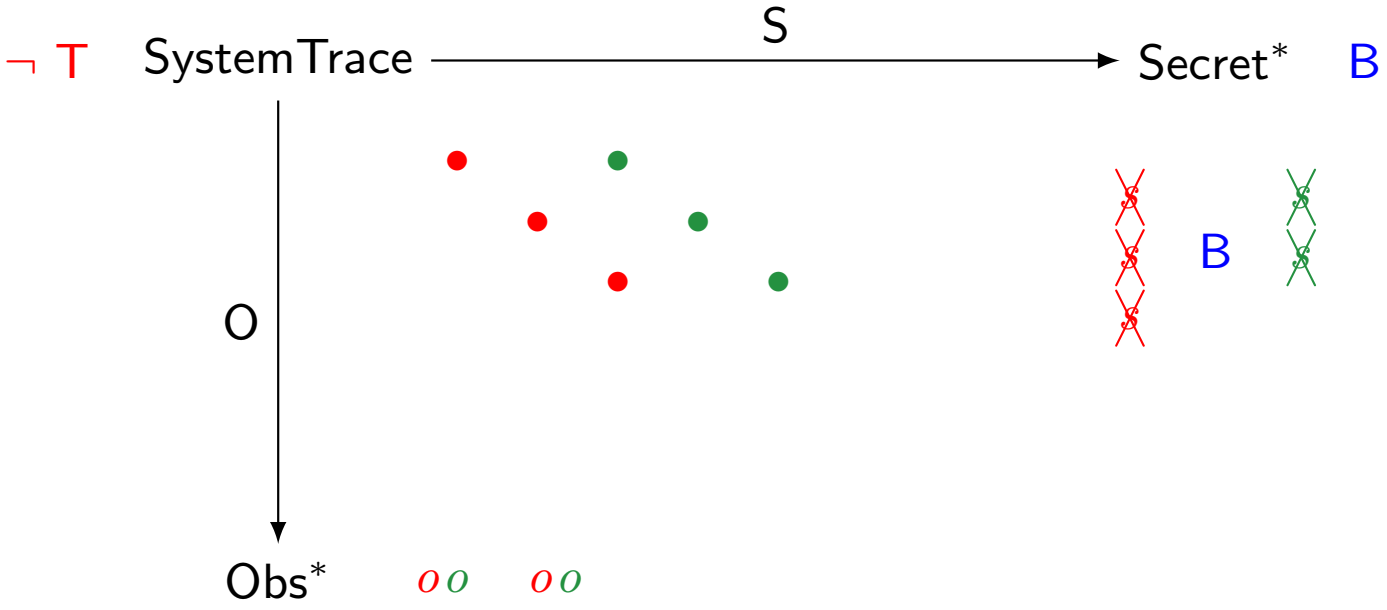
Bounded-Deducibility (BD) Security



Bounded-Deducibility (BD) Security

Proof by unwinding ✓

$$B \mapsto \Delta \subseteq \mathsf{State} \times \mathsf{Secret}^* \times \mathsf{State} \times \mathsf{Secret}^*$$



Proof by Unwinding

Proof by Unwinding

$\Delta \subseteq \mathsf{State} \times \mathsf{Secret}^* \times \mathsf{State} \times \mathsf{Secret}^*$

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

Source	Trigger	Bound
Paper Content	Paper Authorship	Last Uploaded Version
	Paper Authorship or PC Membership^{B}	Nothing
Review	Review Authorship	Last Edited Version Before Discussion and All the Later Versions
	Review Authorship or Non-Conflict PC Membership^{D}	Last Edited Version Before Notification
	Review Authorship or Non-Conflict PC Membership^{D} or PC Membership^{N} or Paper Authorship^{N}	Nothing
Discussion	Non-Conflict PC Membership	Nothing
Decision	Non-Conflict PC Membership	Last Edited Version
	Non-Conflict PC Membership or PC Membership^{N} or Paper Authorship^{N}	Nothing
Reviewer Assignment to Paper	Non-Conflict PC Membership^{R}	Non-Conflict PC Membership of Reviewers

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

Source	Trigger	Bound
Paper Content	Paper Authorship	Last Uploaded Version
	Paper Authorship or PC Membership[B]	Nothing
Review	Review Authorship	Last Edited Version Before Discussion and All the Later Versions
	Review Authorship or Non-Conflict PC Membership[D]	Last Edited Version Before Notification
	Review Authorship or Non-Conflict PC Membership[D] or PC Membership[N] or Paper Authorship[N]	Nothing
Discussion	Non-Conflict PC Membership	Nothing
Decision	Non-Conflict PC Membership	Last Edited Version
	Non-Conflict PC Membership or PC Membership[N] or Paper Authorship[N]	Nothing
Reviewer Assignment to Paper	Non-Conflict PC Membership[R]	Non-Conflict PC Membership of Reviewers

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

Source	Trigger	Bound
Paper Content	Paper Authorship	Last Uploaded Version
	Paper Authorship or PC Membership^{B}	Nothing
Review	Review Authorship	Last Edited Version Before Discussion and All the Later Versions
	Review Authorship or Non-Conflict PC Membership^{D}	Last Edited Version Before Notification
	Review Authorship or Non-Conflict PC Membership^{D} or PC Membership^{N} or Paper Authorship^{N}	Nothing
Discussion	Non-Conflict PC Membership	Nothing
Decision	Non-Conflict PC Membership	Last Edited Version
	Non-Conflict PC Membership or PC Membership^{N} or Paper Authorship^{N}	Nothing
Reviewer Assignment to Paper	Non-Conflict PC Membership^{R}	Non-Conflict PC Membership of Reviewers

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

Source	Trigger	Bound
Paper Content	Paper Authorship	Last Uploaded Version
	Paper Authorship or PC Membership^{B}	Nothing
Review	Review Authorship	Last Edited Version Before Discussion and All the Later Versions
	Review Authorship or Non-Conflict PC Membership^{D}	Last Edited Version Before Notification
	Review Authorship or Non-Conflict PC Membership^{D} or PC Membership^{N} or Paper Authorship^{N}	Nothing
Discussion	Non-Conflict PC Membership	Nothing
Decision	Non-Conflict PC Membership	Last Edited Version
	Non-Conflict PC Membership or PC Membership^{N} or Paper Authorship^{N}	Nothing
Reviewer Assignment to Paper	Non-Conflict PC Membership^{R}	Non-Conflict PC Membership of Reviewers

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

Source	Trigger	Bound
Paper Content	Paper Authorship	Last Uploaded Version
	Paper Authorship or PC Membership^{B}	Nothing
Review	Review Authorship	Last Edited Version Before Discussion and All the Later Versions
	Review Authorship or Non-Conflict PC Membership^{D}	Last Edited Version Before Notification
	Review Authorship or Non-Conflict PC Membership^{D} or PC Membership^{N} or Paper Authorship^{N}	Nothing
Discussion	Non-Conflict PC Membership	Nothing
Decision	Non-Conflict PC Membership	Last Edited Version
	Non-Conflict PC Membership or PC Membership^{N} or Paper Authorship^{N}	Nothing
Reviewer Assignment to Paper	Non-Conflict PC Membership^{R}	Non-Conflict PC Membership of Reviewers

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

Source	Trigger	Bound
Paper Content	Paper Authorship	Last Uploaded Version
	Paper Authorship or PC Membership^{B}	Nothing
Review	Review Authorship	Last Edited Version Before Discussion and All the Later Versions
	Review Authorship or Non-Conflict PC Membership^{D}	Last Edited Version Before Notification
	Review Authorship or Non-Conflict PC Membership^{D} or PC Membership^{N} or Paper Authorship^{N}	Nothing
Discussion	Non-Conflict PC Membership	Nothing
Decision	Non-Conflict PC Membership	Last Edited Version
	Non-Conflict PC Membership or PC Membership^{N} or Paper Authorship^{N}	Nothing
Reviewer Assignment to Paper	Non-Conflict PC Membership^{R}	Non-Conflict PC Membership of Reviewers

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

Source	Trigger	Bound
Paper Content	Paper Authorship	Last Uploaded Version
	Paper Authorship or PC Membership[B]	Nothing
Review	Review Authorship	Last Edited Version Before Discussion and All the Later Versions
	Review Authorship or Non-Conflict PC Membership[D]	Last Edited Version Before Notification
	Review Authorship or Non-Conflict PC Membership[D] or PC Membership[N] or Paper Authorship[N]	Nothing
Discussion	Non-Conflict PC Membership	Nothing
Decision	Non-Conflict PC Membership	Last Edited Version
	Non-Conflict PC Membership or PC Membership[N] or Paper Authorship[N]	Nothing
Reviewer Assignment to Paper	Non-Conflict PC Membership[R]	Non-Conflict PC Membership of Reviewers

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

Source	Trigger	Bound
Paper Content	Paper Authorship	Last Uploaded Version
	Paper Authorship or PC Membership^{B}	Nothing
Review	Review Authorship	Last Edited Version Before Discussion and All the Later Versions
	Review Authorship or Non-Conflict PC Membership^{D}	Last Edited Version Before Notification
	Review Authorship or Non-Conflict PC Membership^{D} or PC Membership^{N} or Paper Authorship^{N}	Nothing
Discussion	Non-Conflict PC Membership	Nothing
Decision	Non-Conflict PC Membership	Last Edited Version
	Non-Conflict PC Membership or PC Membership^{N} or Paper Authorship^{N}	Nothing
Reviewer Assignment to Paper	Non-Conflict PC Membership^{R}	Non-Conflict PC Membership of Reviewers and Number of Reviewers

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

Source	Trigger	Bound
Paper Content	Paper Authorship	Last Uploaded Version
	Paper Authorship or PC Membership^{B}	Nothing
Review	Review Authorship	Last Edited Version Before Discussion and All the Later Versions
	Review Authorship or Non-Conflict PC Membership^{D}	Last Edited Version Before Notification
	Review Authorship or Non-Conflict PC Membership^{D} or PC Membership^{N} or Paper Authorship^{N}	Nothing
Discussion	Non-Conflict PC Membership	Nothing
Decision	Non-Conflict PC Membership	Last Edited Version
	Non-Conflict PC Membership or PC Membership^{N} or Paper Authorship^{N}	Nothing
Reviewer Assignment to Paper	Non-Conflict PC Membership^{R}	Non-Conflict PC Membership of Reviewers and Number of Reviewers
	Non-Conflict PC Membership^{R} or Paper Authorship^{N}	Non-Conflict PC Membership of Reviewers

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

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

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) $\longrightarrow$ 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

Accountability

We proved: one cannot learn beyond such an such unless one is or becomes such and such

Accountability

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  ?

Accountability

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  ?

 has registered 

Accountability

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  ?

 has registered 

$\Downarrow$

Accountability

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  ?

 has registered 

$\Downarrow$

$\vdots$

$\Downarrow$

Accountability

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  ?

 has registered 

$\Downarrow$

$\vdots$

$\Downarrow$

Confidentiality + Safety + Accountability $\Longrightarrow$ Relax

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)

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."

What I Like About CoCon

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

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

And practical considerations about CoCon keep bringing new theoretical challenges...

CoCon at TABLEAUX 2015?

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?

Hans de Nivelle
conference chair

CoCon? Not a chance.

But maybe CoCon++...

CoCon++

CoCon plus the following features:

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++

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
  - ...

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

CoCon++

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
  - ...

CoCon++

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
  - ...

CoCon++

Various convenience listings, e.g.:

For PC members: papers listed by average score

CoCon++

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.

CoCon++

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.

TABLEAUX
PC member in
Discussion phase

CoCon++

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.

TABLEAUX
PC member in
Discussion phase

I'm also an author and I see my paper
listed somewhere in the middle.

CoCon++

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.

TABLEAUX
PC member in
Discussion 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?

CoCon++

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.

TABLEAUX
PC member in
Discussion 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.

What had happened?

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.

What had happened?

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

What had happened?

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

What had happened?

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

Additional channels

Email notifications:

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

Additional channels

Email notifications:

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

Additional channels

Email notifications:

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

Additional channels

Email notifications:

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

User at CoCon testing session

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

Additional channels

Email notifications:

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

User at CoCon testing session

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.

General Pattern

Given a confidentiality-verified system

Extend the system

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

General Pattern

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

General Pattern

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

General Pattern

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!

General Pattern

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++...

Example for CoCon++

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

Example for CoCon++

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

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

Example for CoCon++

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

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

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

Example for CoCon++

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

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

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

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

Example for CoCon++

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

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

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

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

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

Example for CoCon++

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

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

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

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

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

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

Example for CoCon++

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

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

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

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

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

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

Final output of listPapersSorted(U): $(r_1 \dots r_m,\ q_1 \dots q_n)$

Example for CoCon++

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)

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

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

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

Final output of 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 listPapersSorted(U) that lists all papers sorted by score.

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

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

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

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

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

Final output of listPapersSorted(U): $(r_1 \dots r_m,\ q_1 \dots q_n)$

Summary: old action invocation + some output filtering

General Pattern

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?

General Pattern

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.

Counterexample

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

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

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

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

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.

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.
Then secret i is revealed.

General Pattern

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.

General Pattern

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.

General Pattern

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?

General Pattern

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?

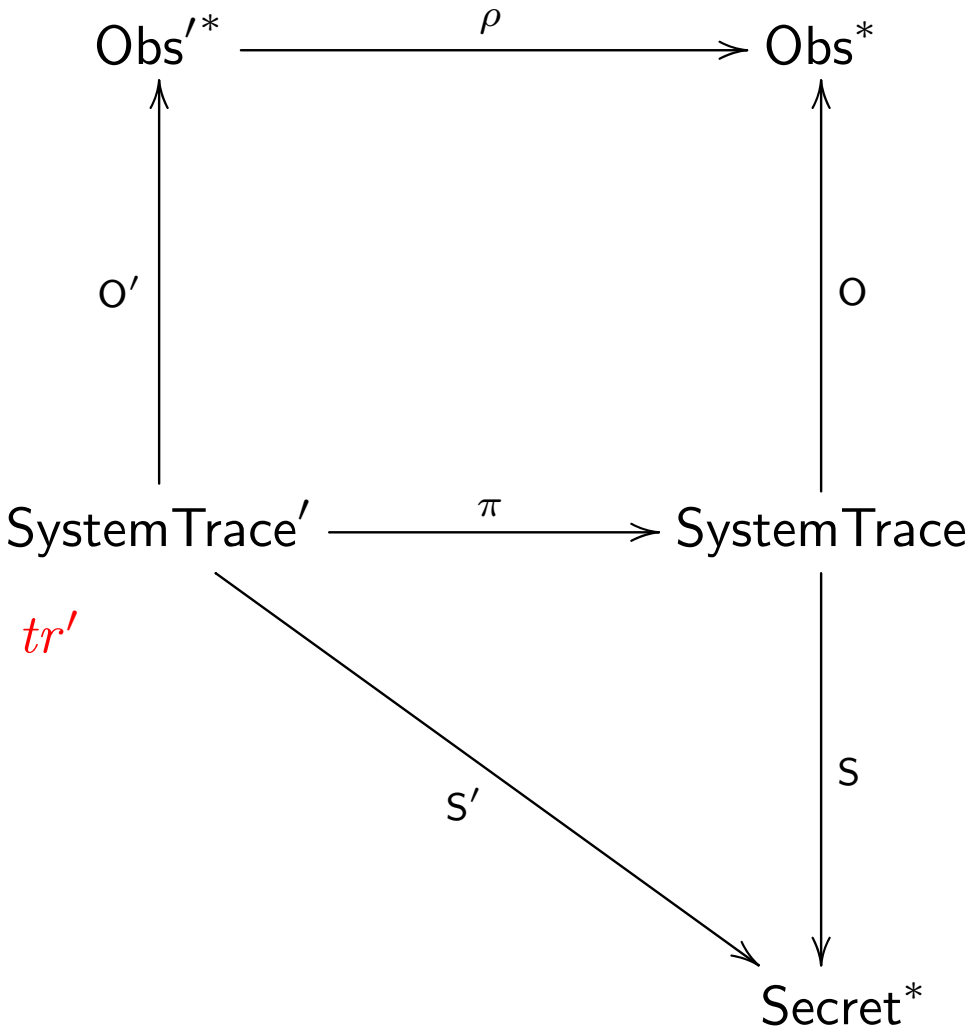
In general, NO. But for CoCon++ done carefully, probably YES.

Why? And can we give the answer for a general class of systems?

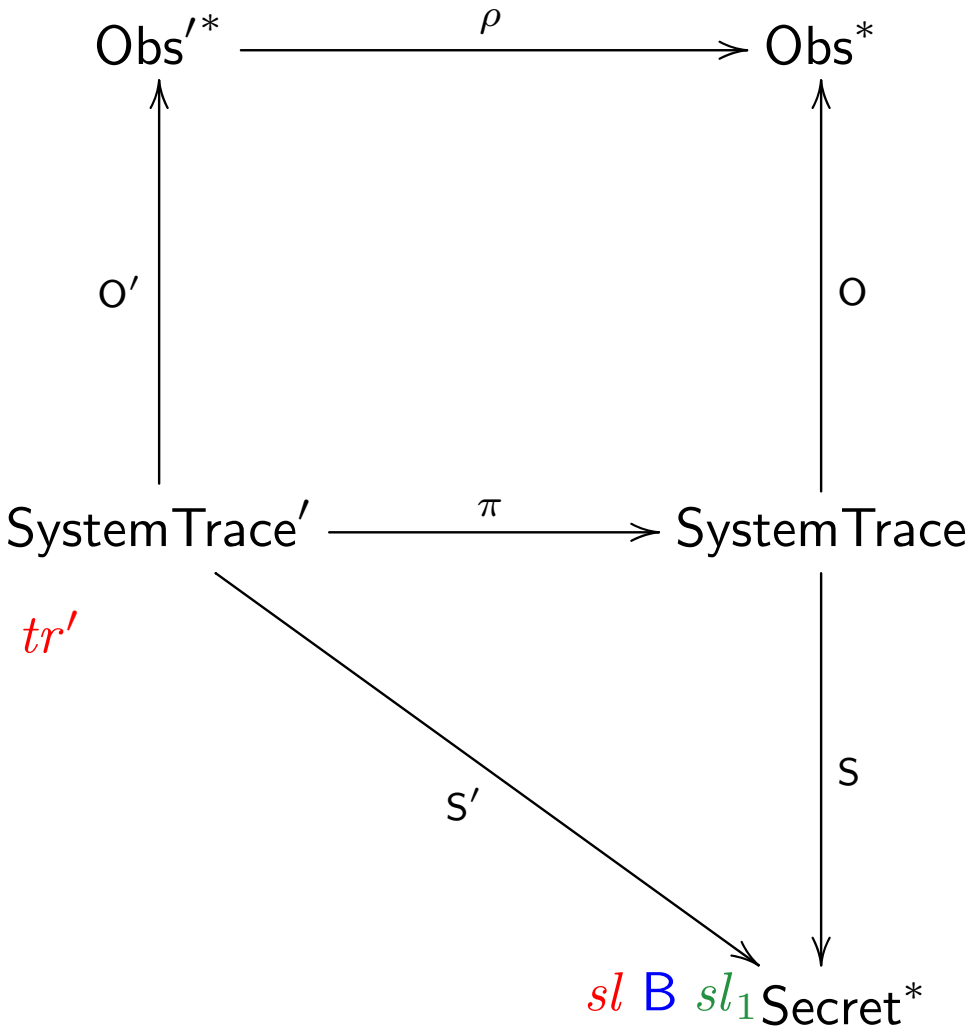
Borrowing BD Security in an Ideal World

$$
\begin{tikzcd}
\mathsf{Obs}'^{*} \arrow[r, "\rho"] & \mathsf{Obs}^{*} \\
\mathsf{SystemTrace}' \arrow[u, "\mathsf{o}'"] \arrow[r, "\pi"] \arrow[dr, "\mathsf{s}'"'] & \mathsf{SystemTrace} \arrow[u, "\mathsf{o}"'] \arrow[d, "\mathsf{s}"] \\
& \mathsf{Secret}^{*}
\end{tikzcd}
$$

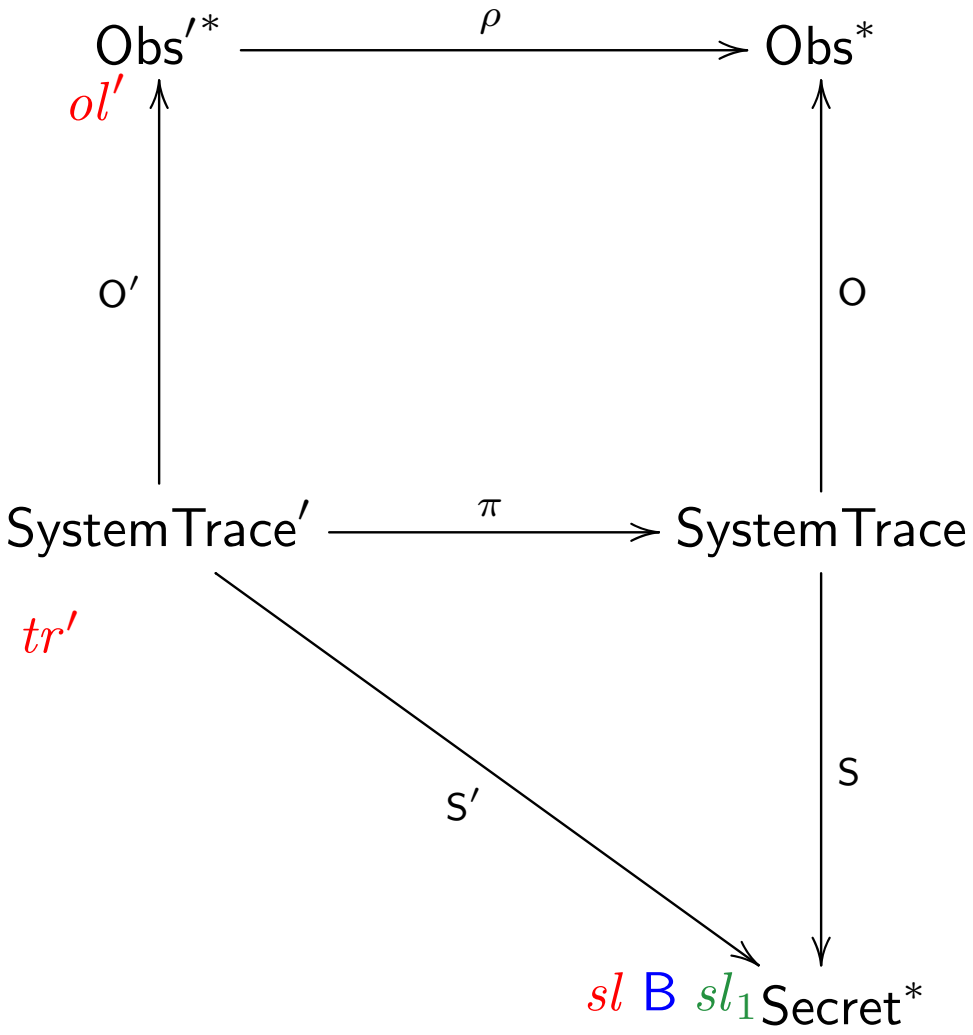
Borrowing BD Security in an Ideal World



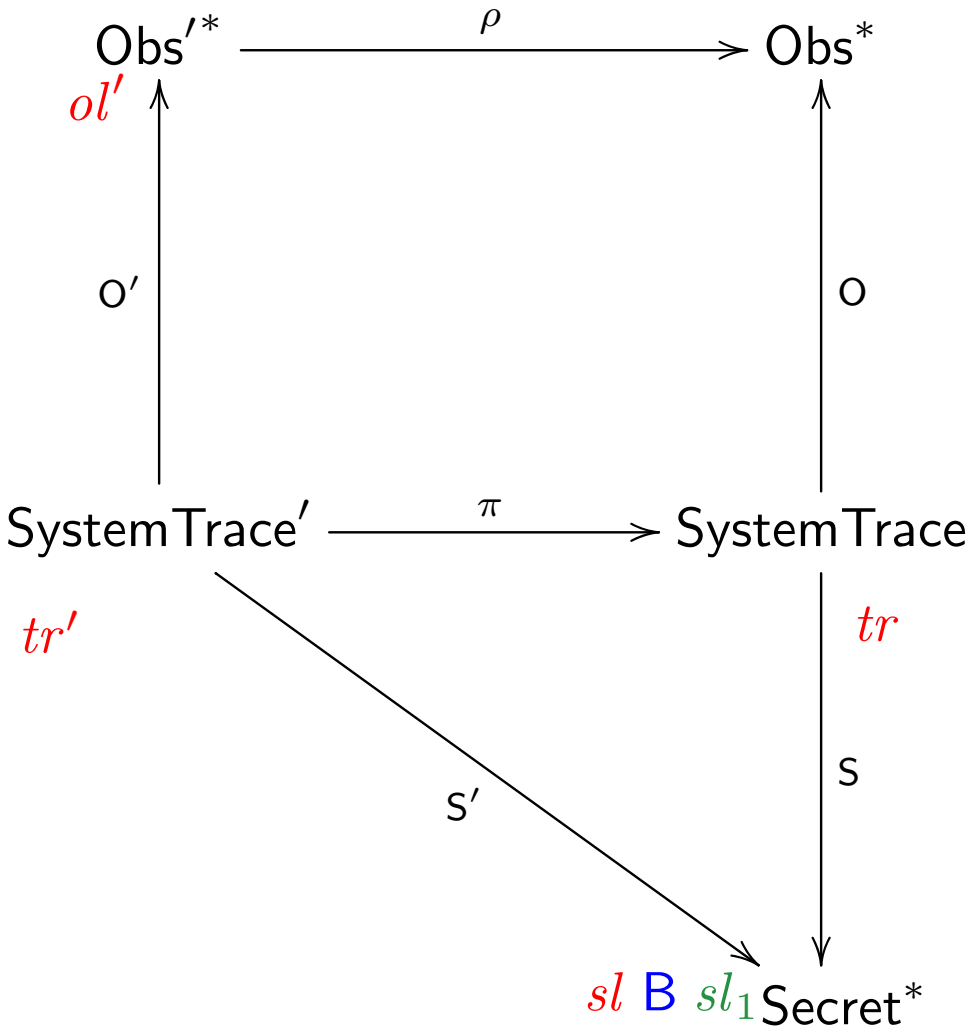
Borrowing BD Security in an Ideal World



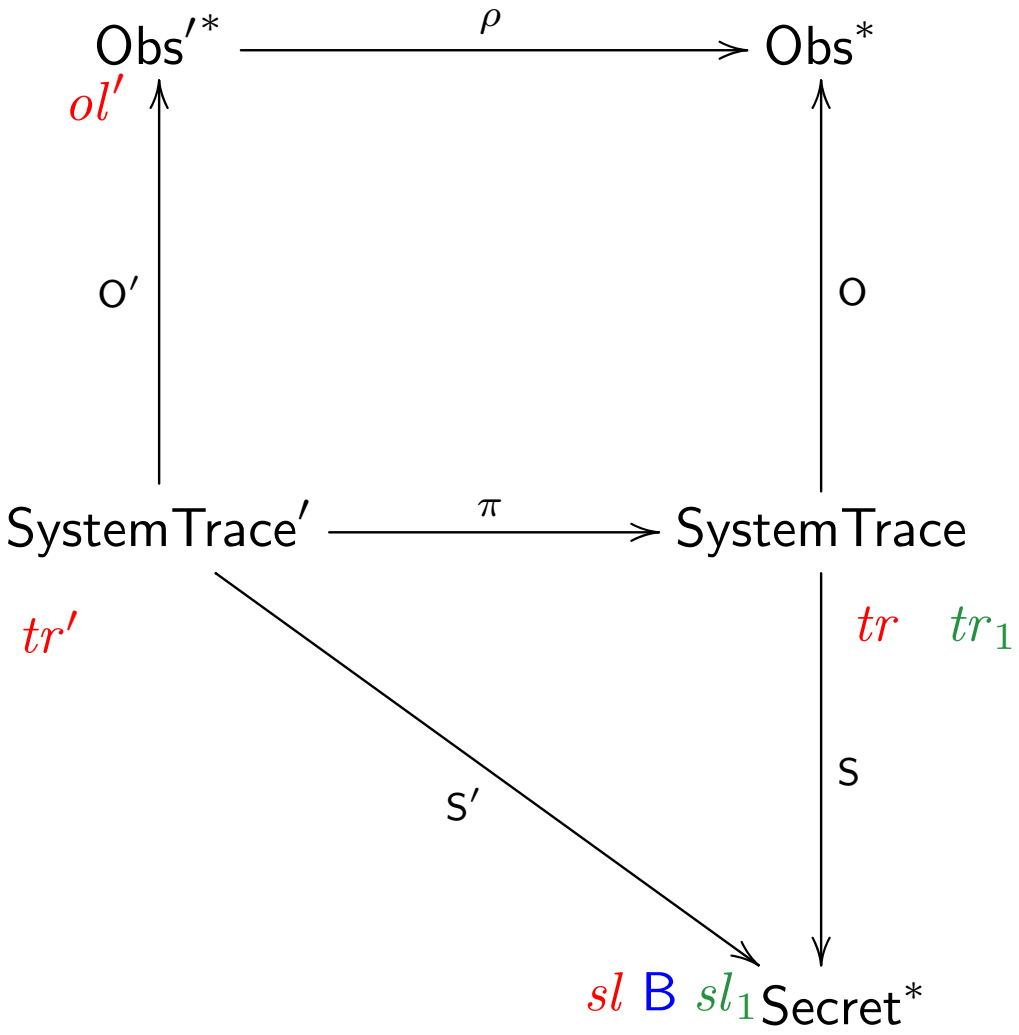
Borrowing BD Security in an Ideal World



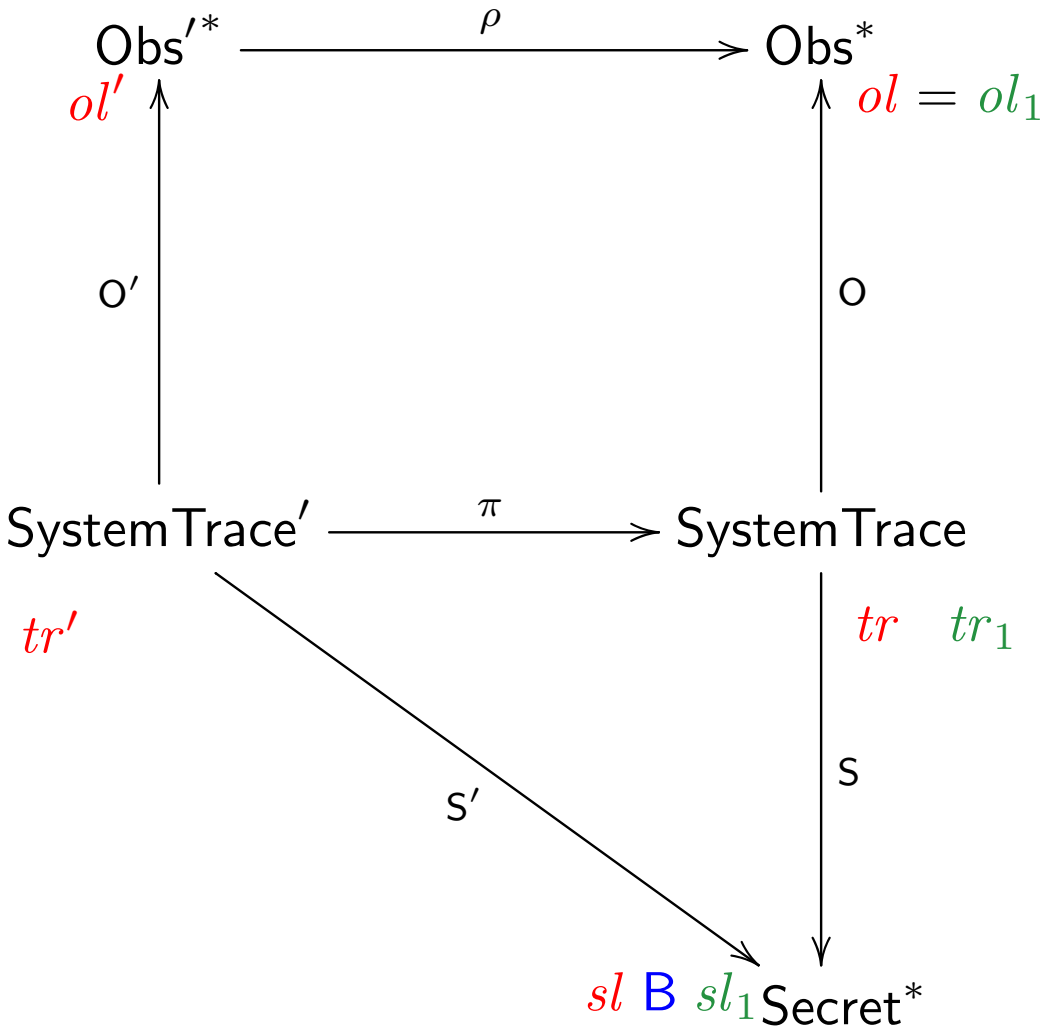
Borrowing BD Security in an Ideal World



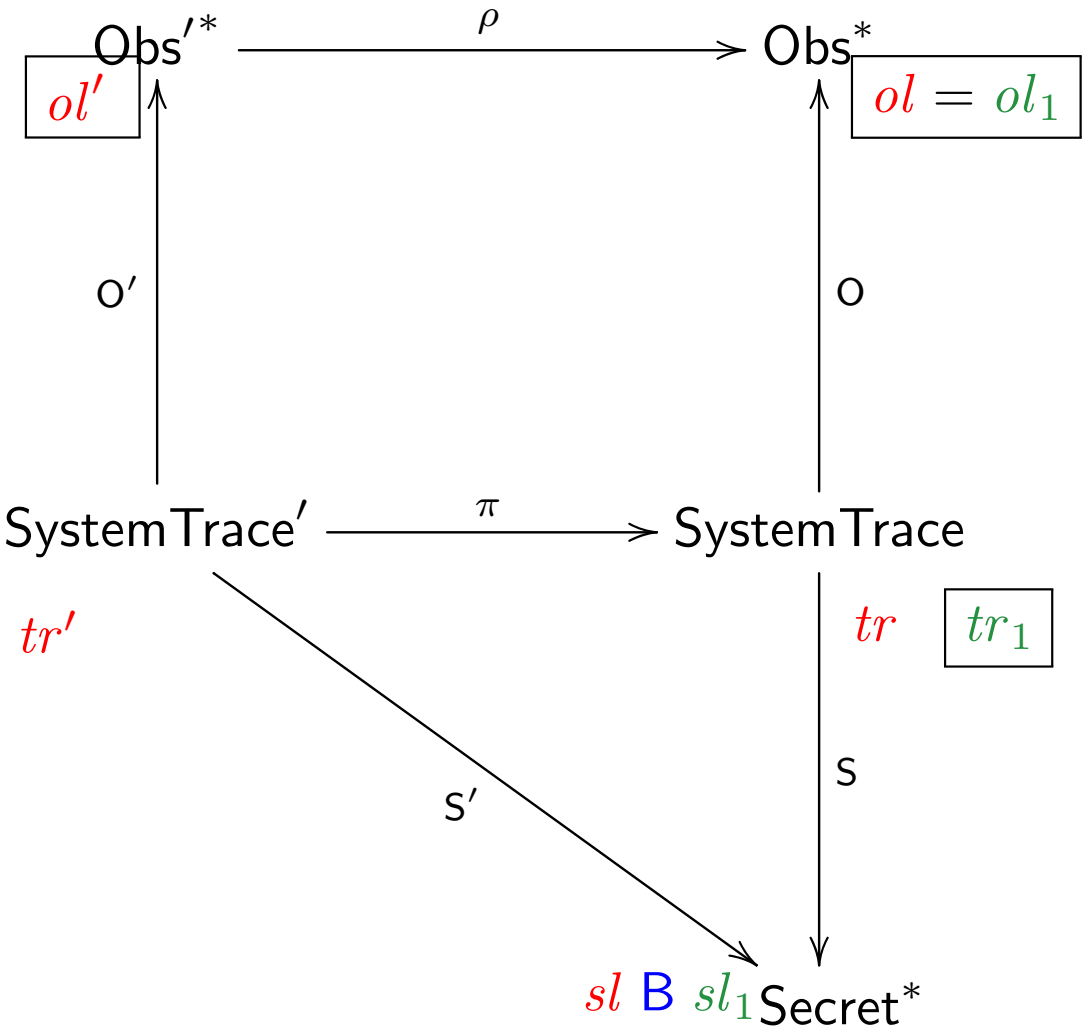
Borrowing BD Security in an Ideal World



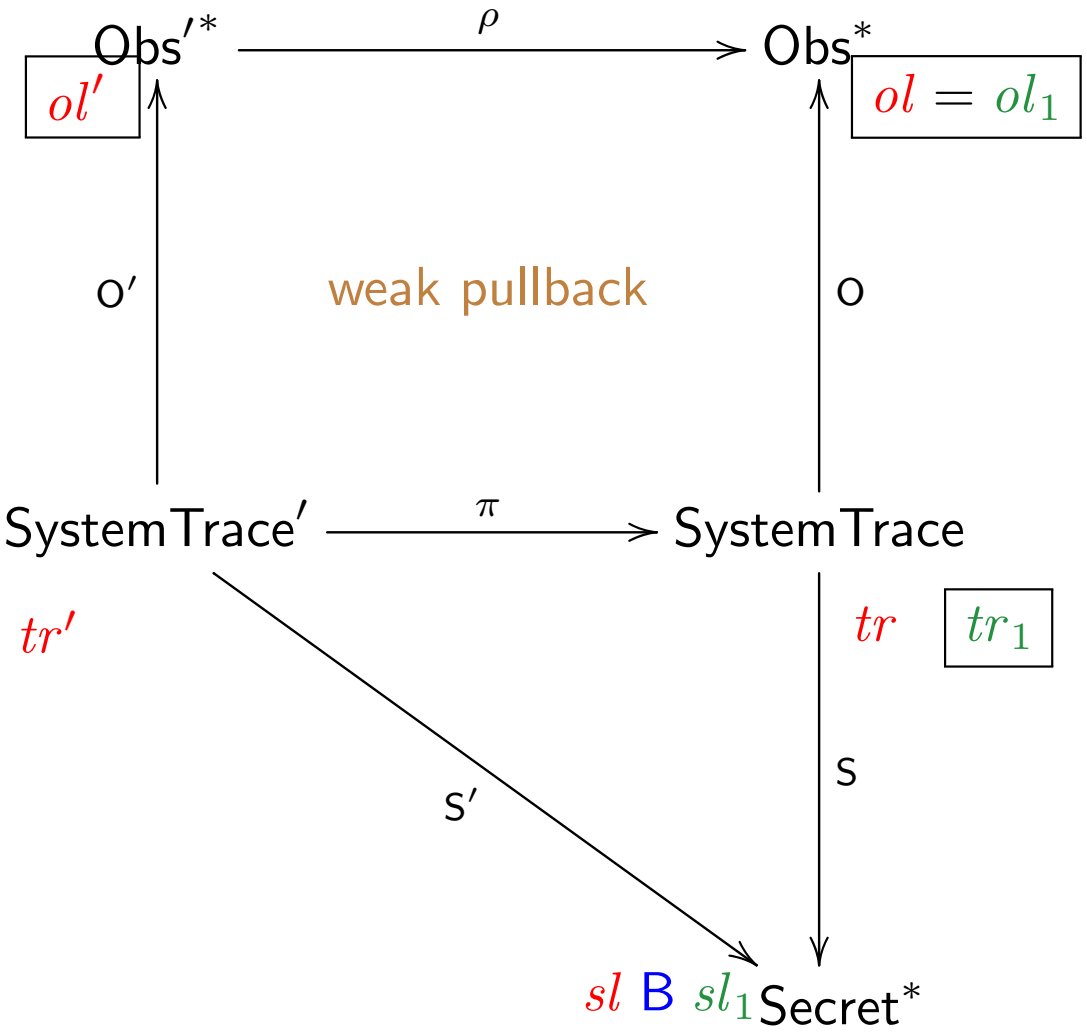
Borrowing BD Security in an Ideal World



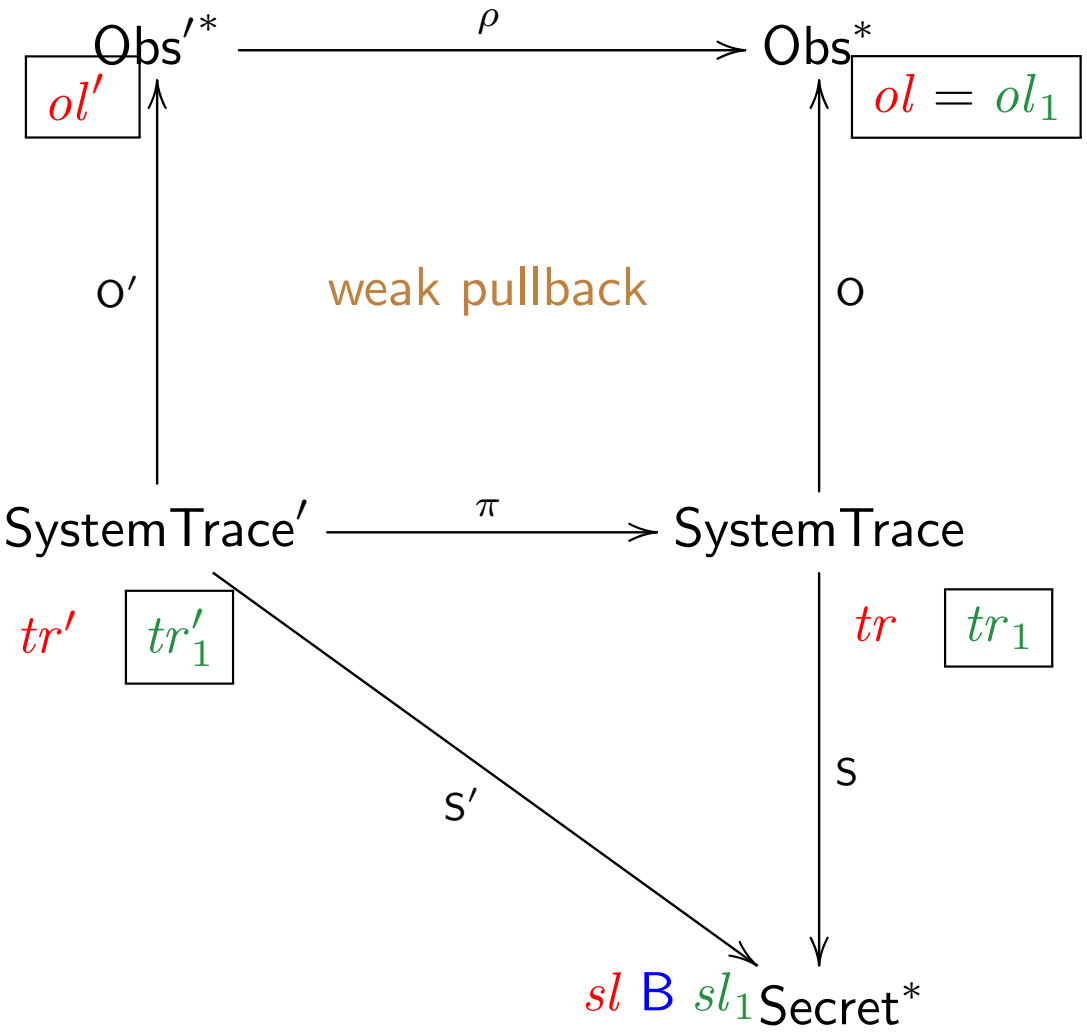
Borrowing BD Security in an Ideal World



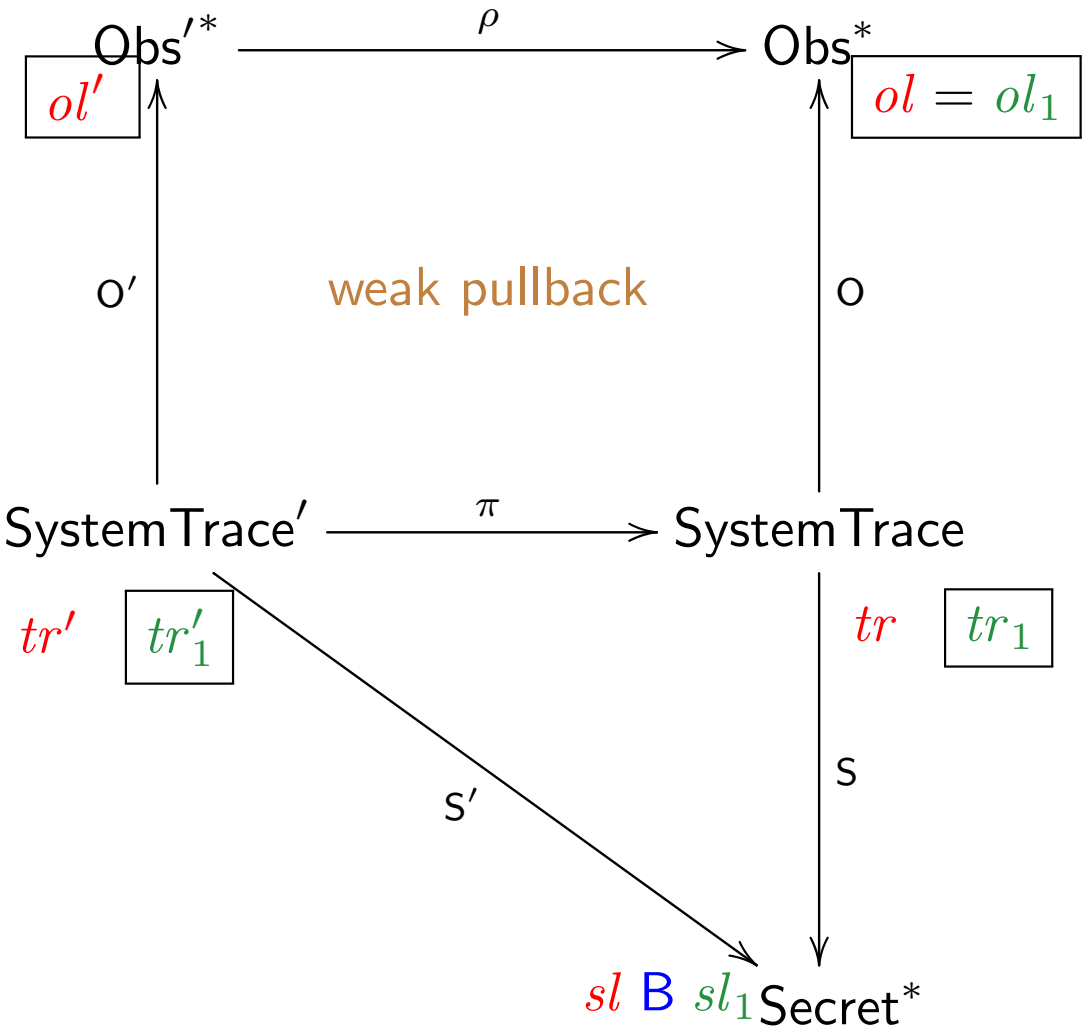
Borrowing BD Security in an Ideal World



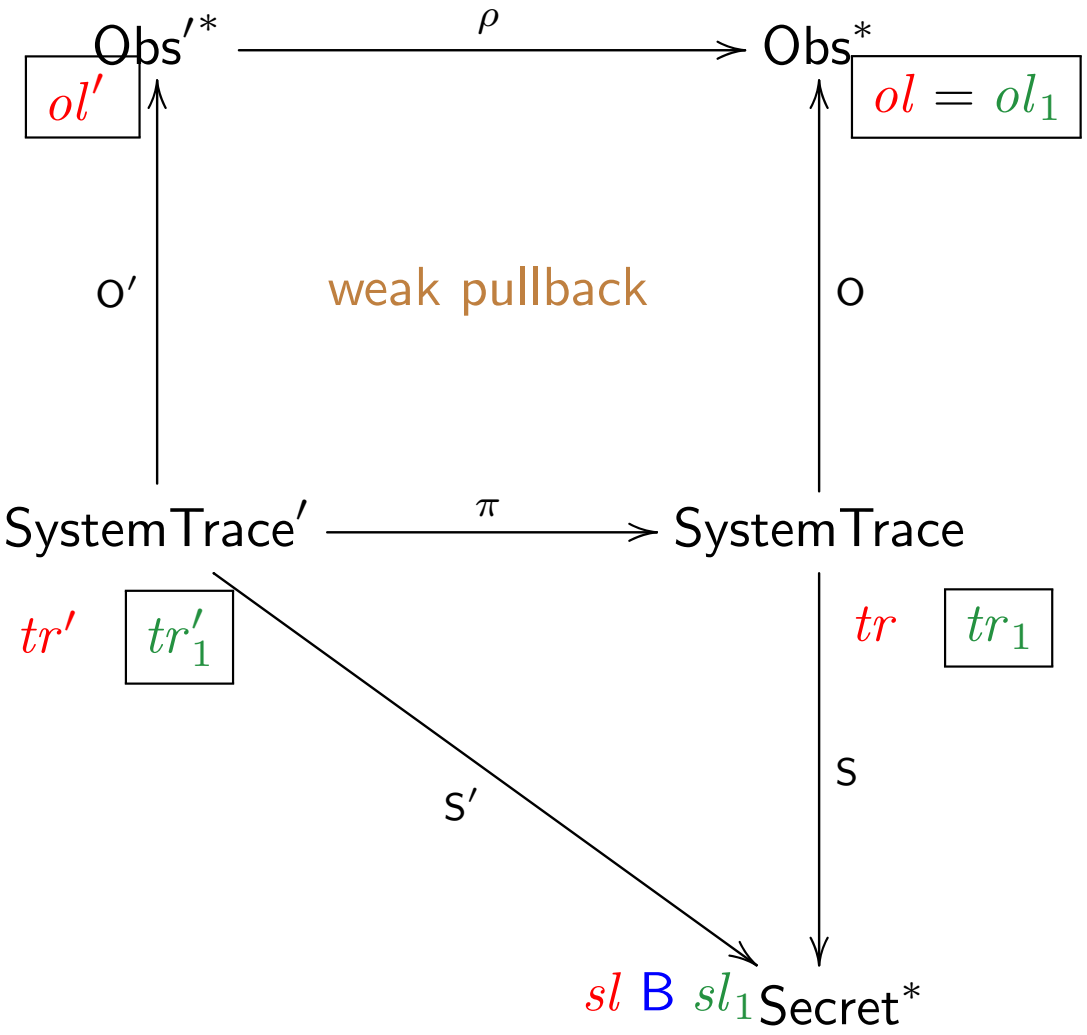
Borrowing BD Security in an Ideal World



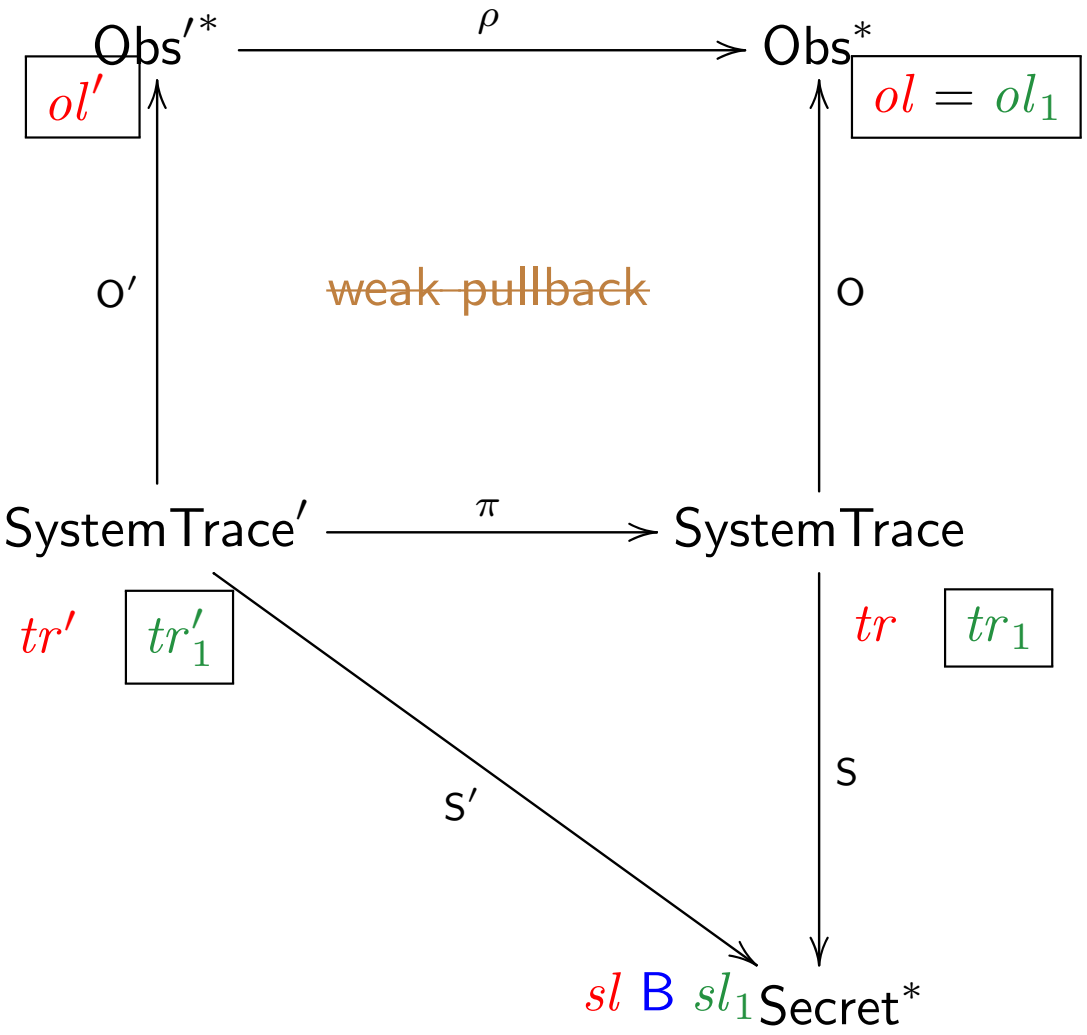
Borrowing BD Security in an Ideal World



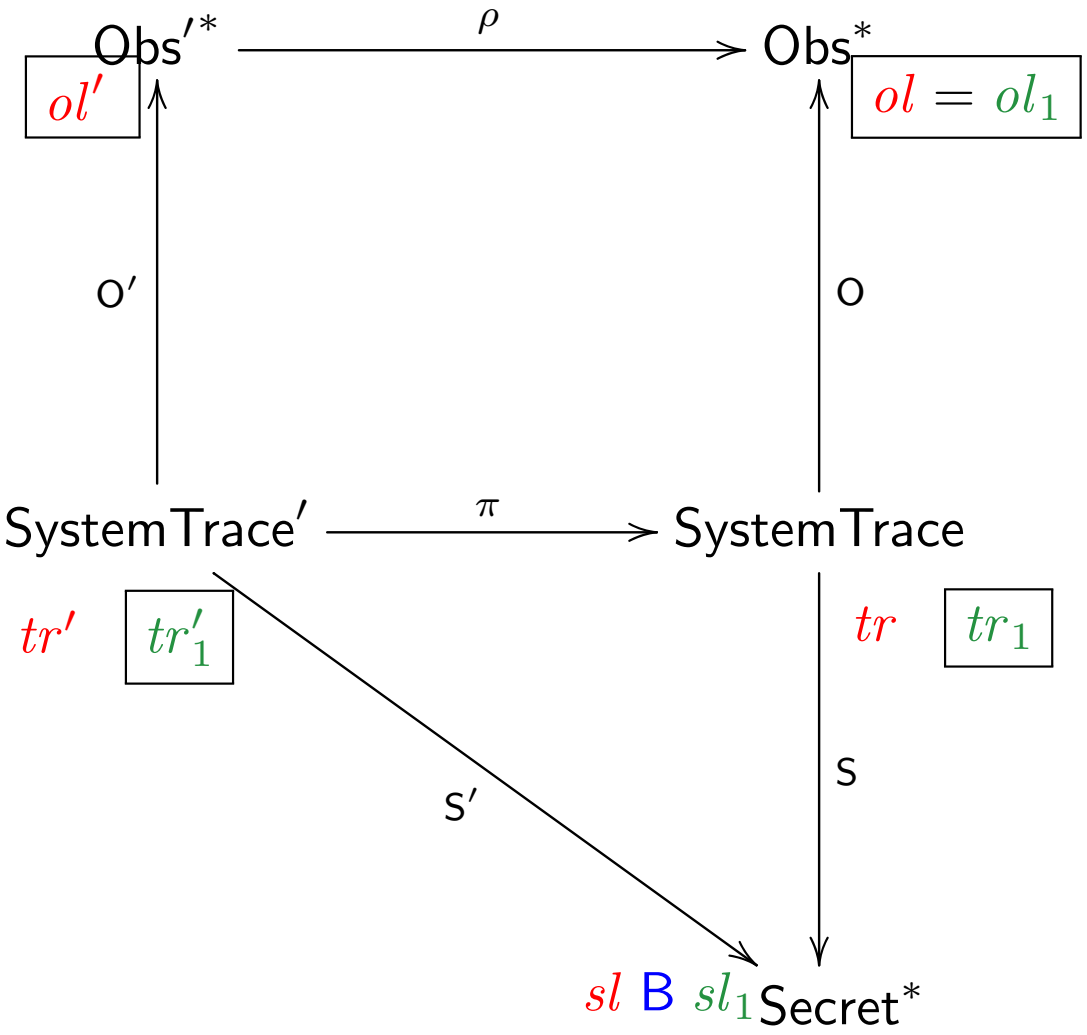
Borrowing BD Security in a Real World



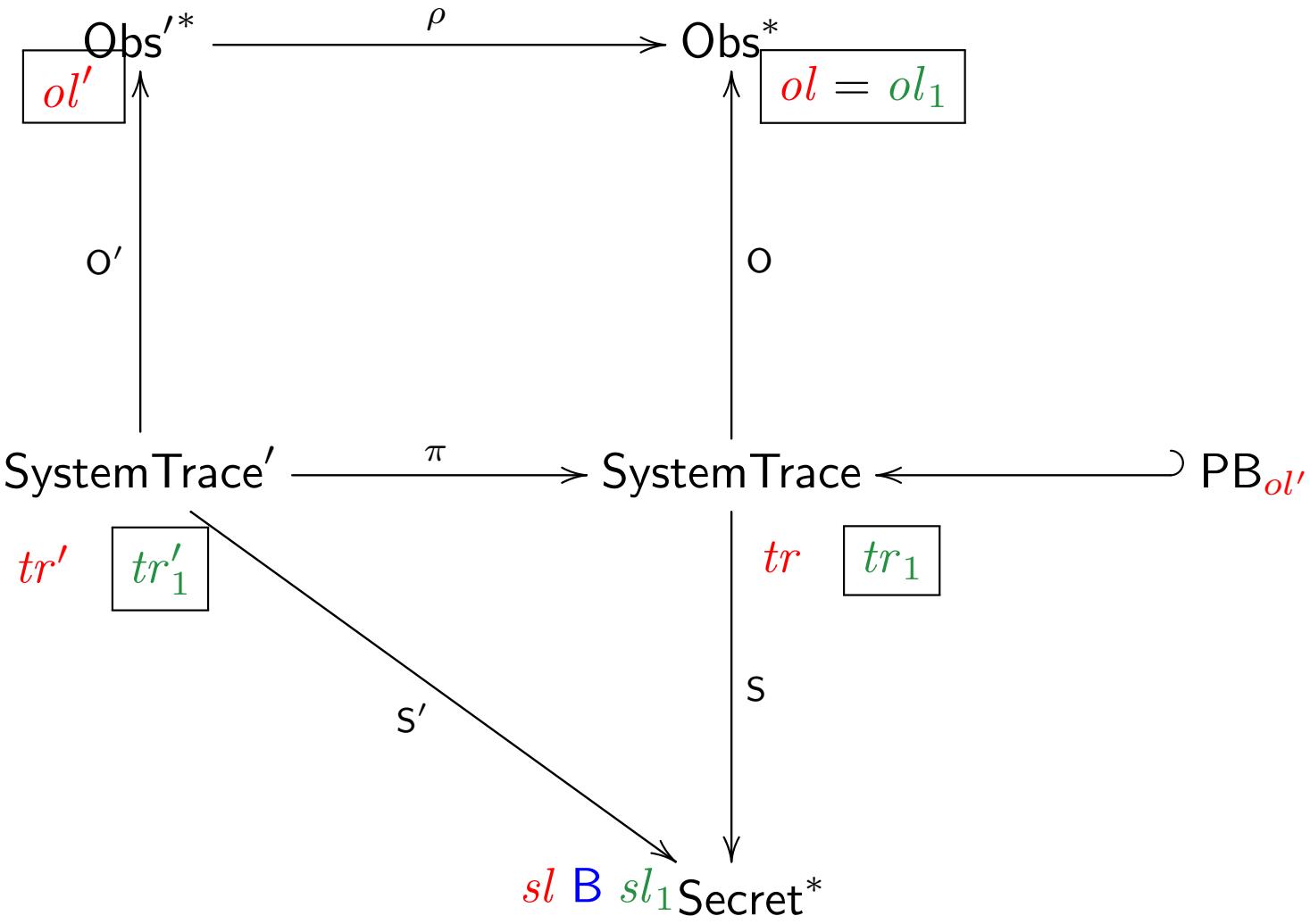
Borrowing BD Security in a Real World



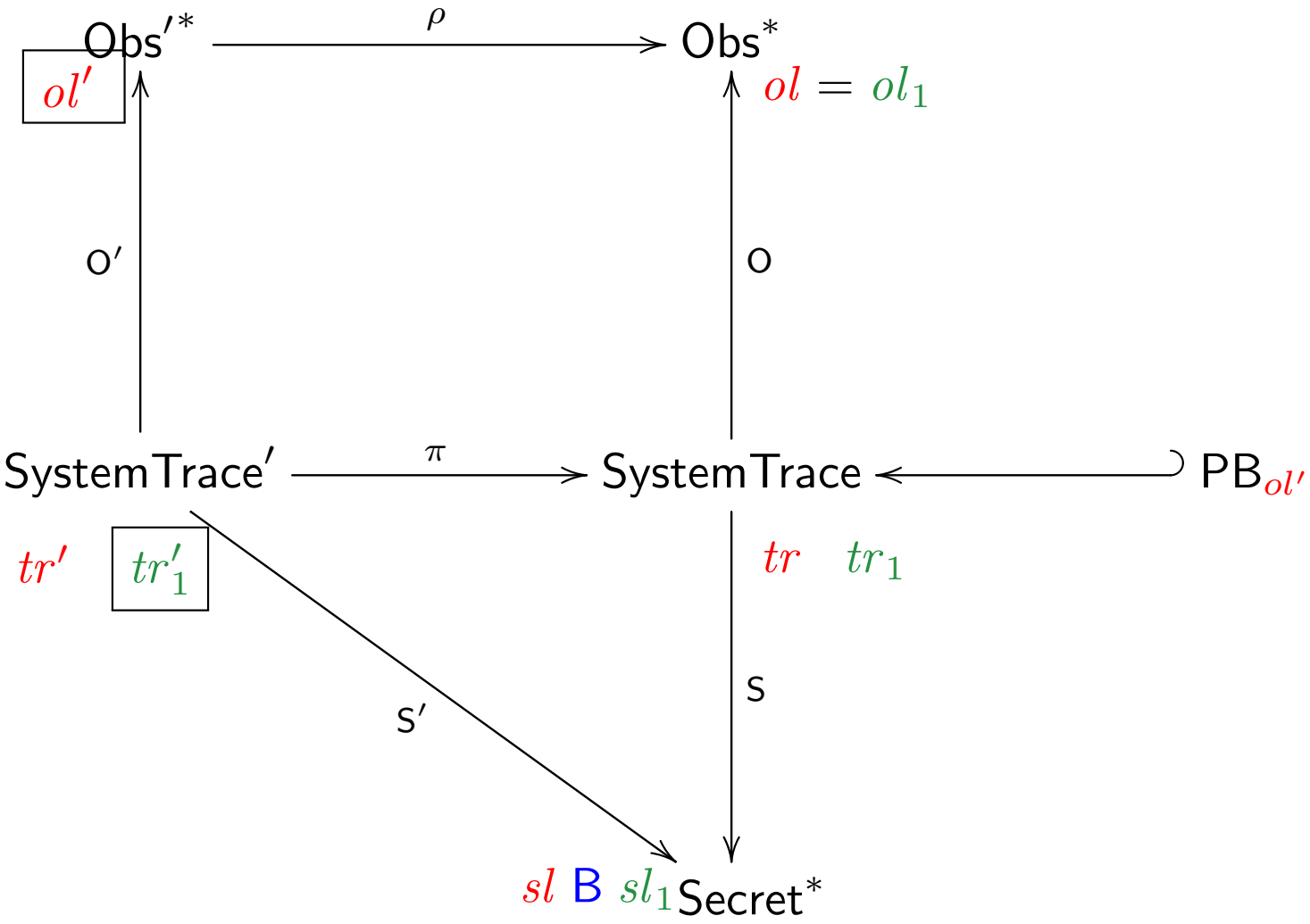
Borrowing BD Security in a Real World



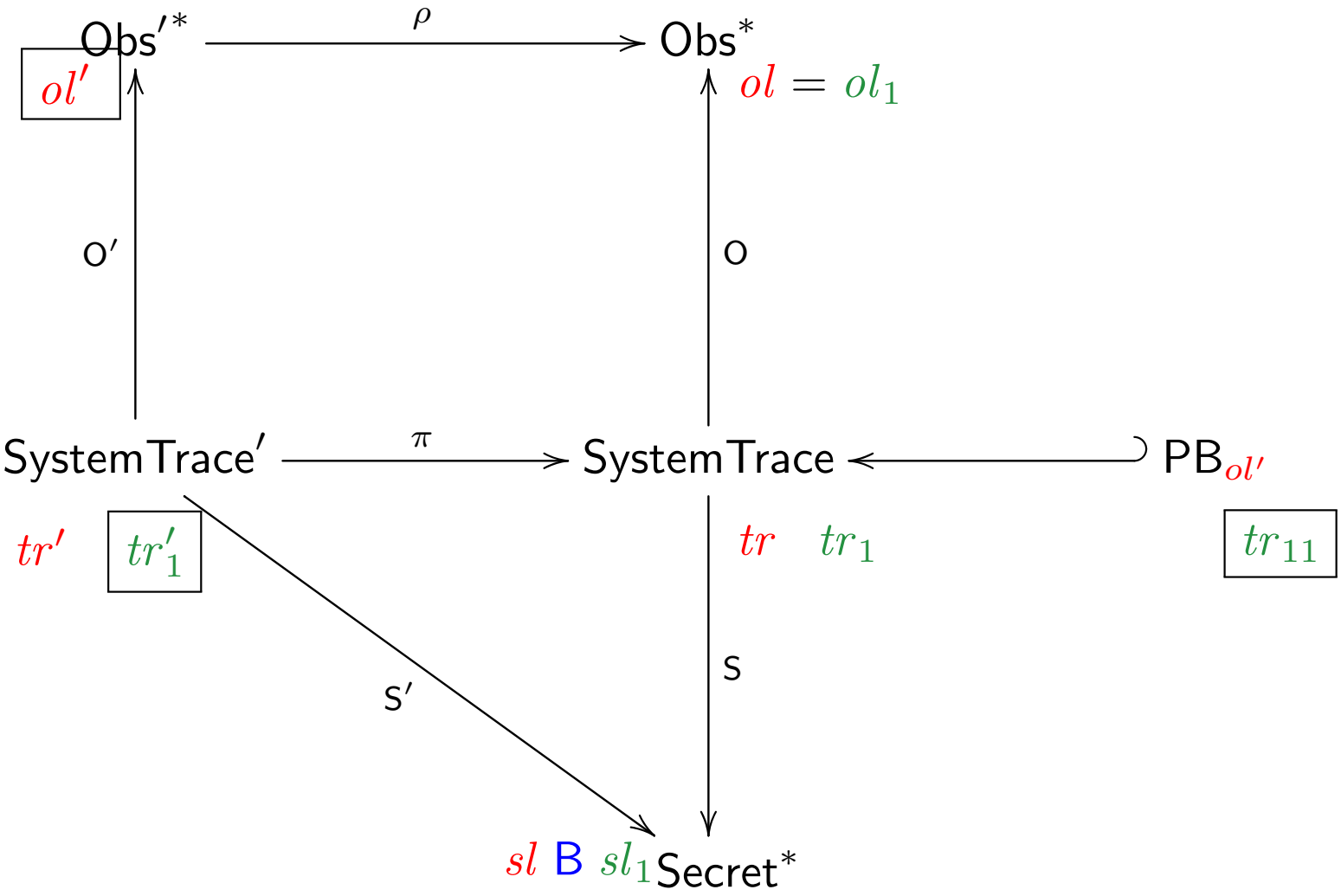
Borrowing BD Security in a Real World



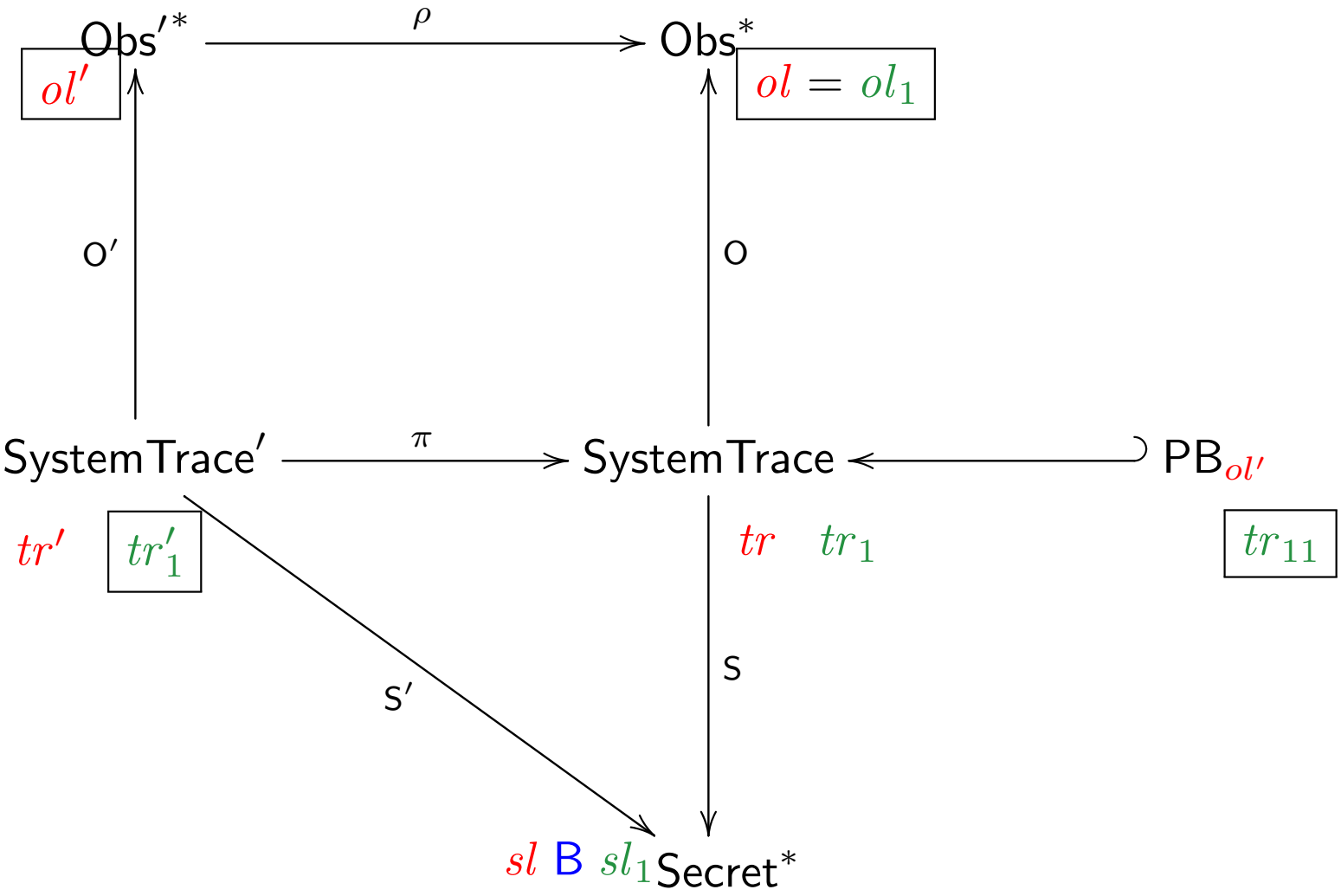
Borrowing BD Security in a Real World



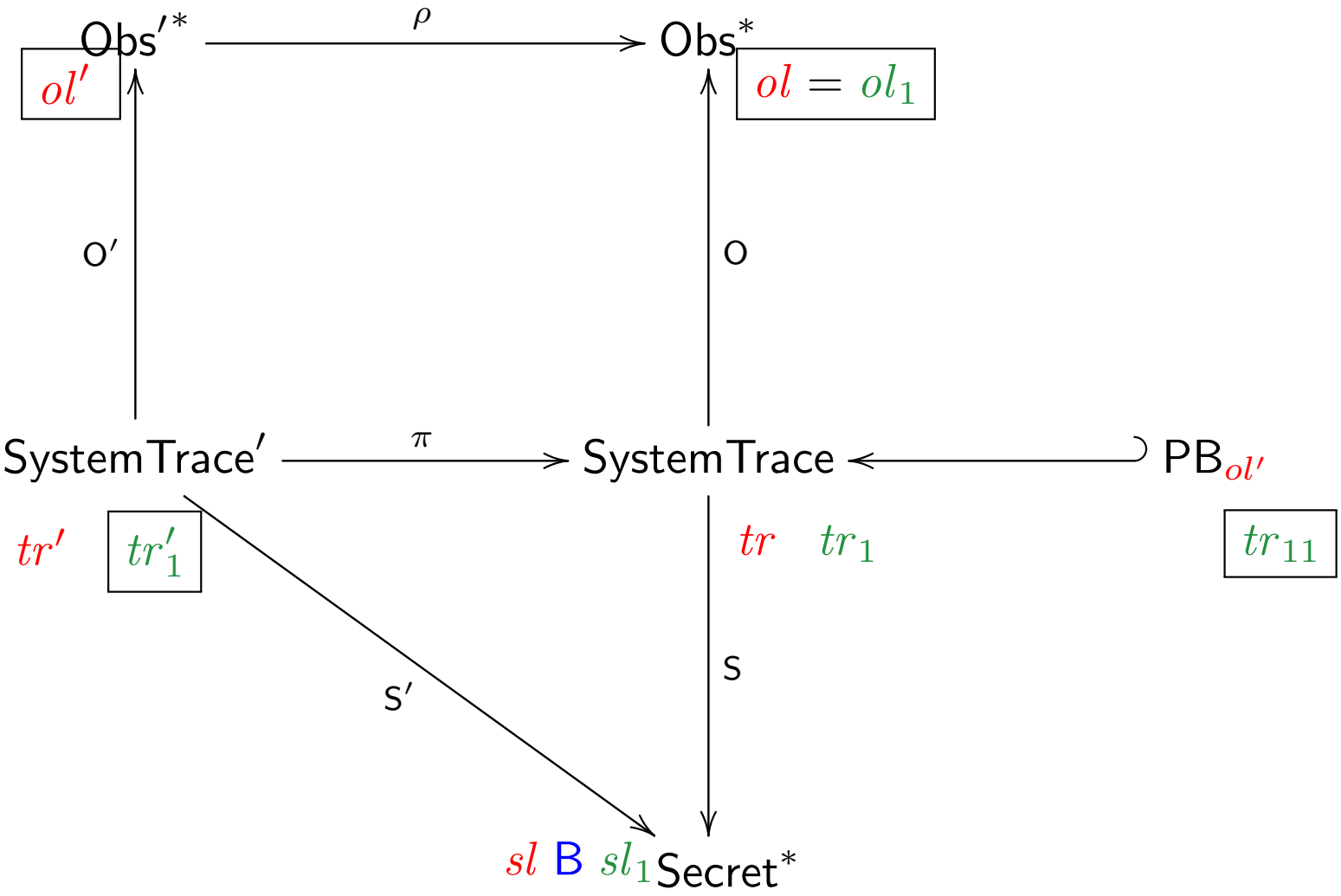
Borrowing BD Security in a Real World



Borrowing BD Security in a Real World



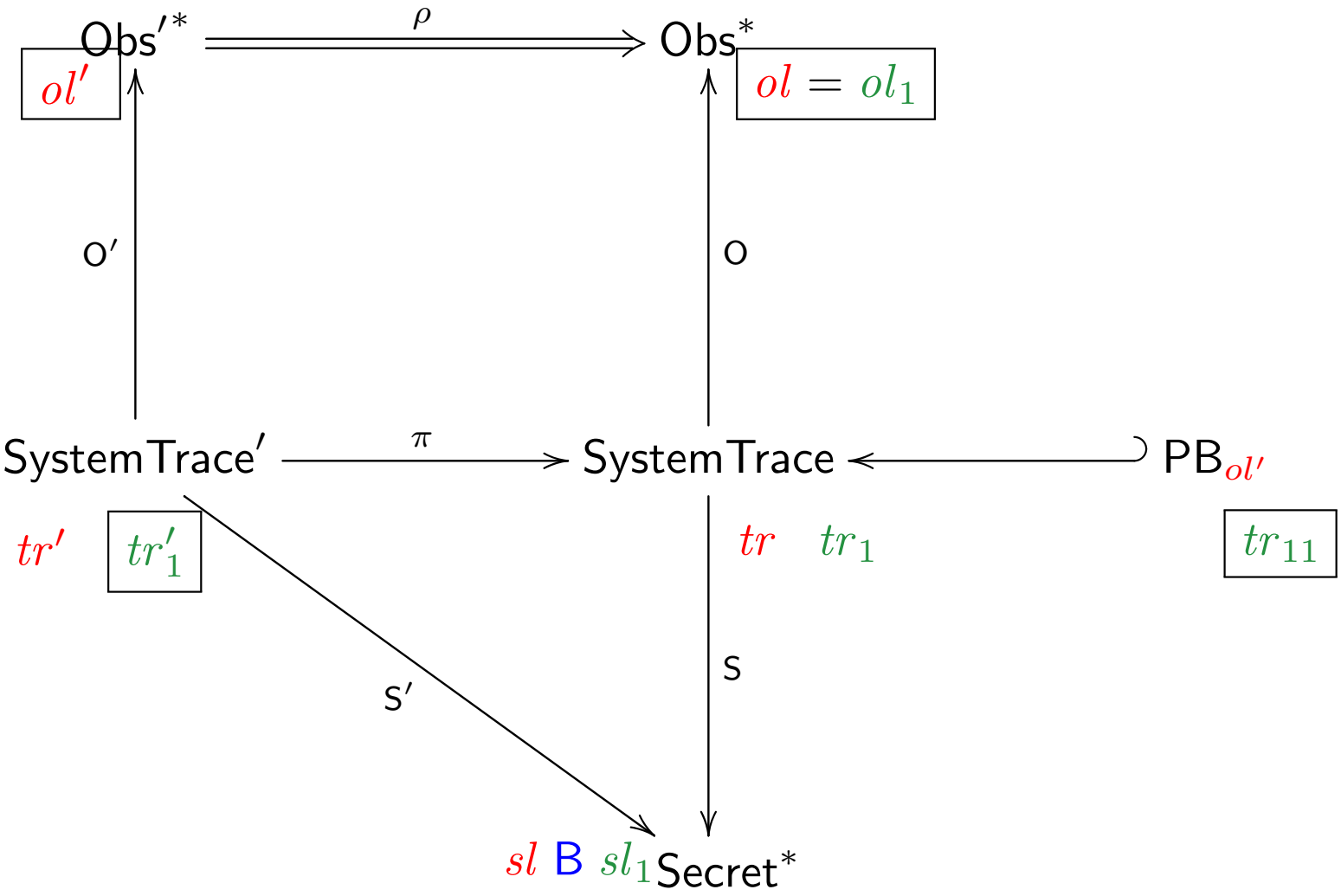
Borrowing BD Security in a Real World



Adjustment property:

$$\forall tr_1, ol'.\ \mathsf{O}\ tr_1 = \rho\ ol' \longrightarrow \exists tr_{11} \in \mathsf{PB}_{ol'}.\ \mathsf{O}\ tr_{11} = \mathsf{O}\ tr_1 \wedge \mathsf{S}\ tr_{11} = \mathsf{S}\ tr_1$$

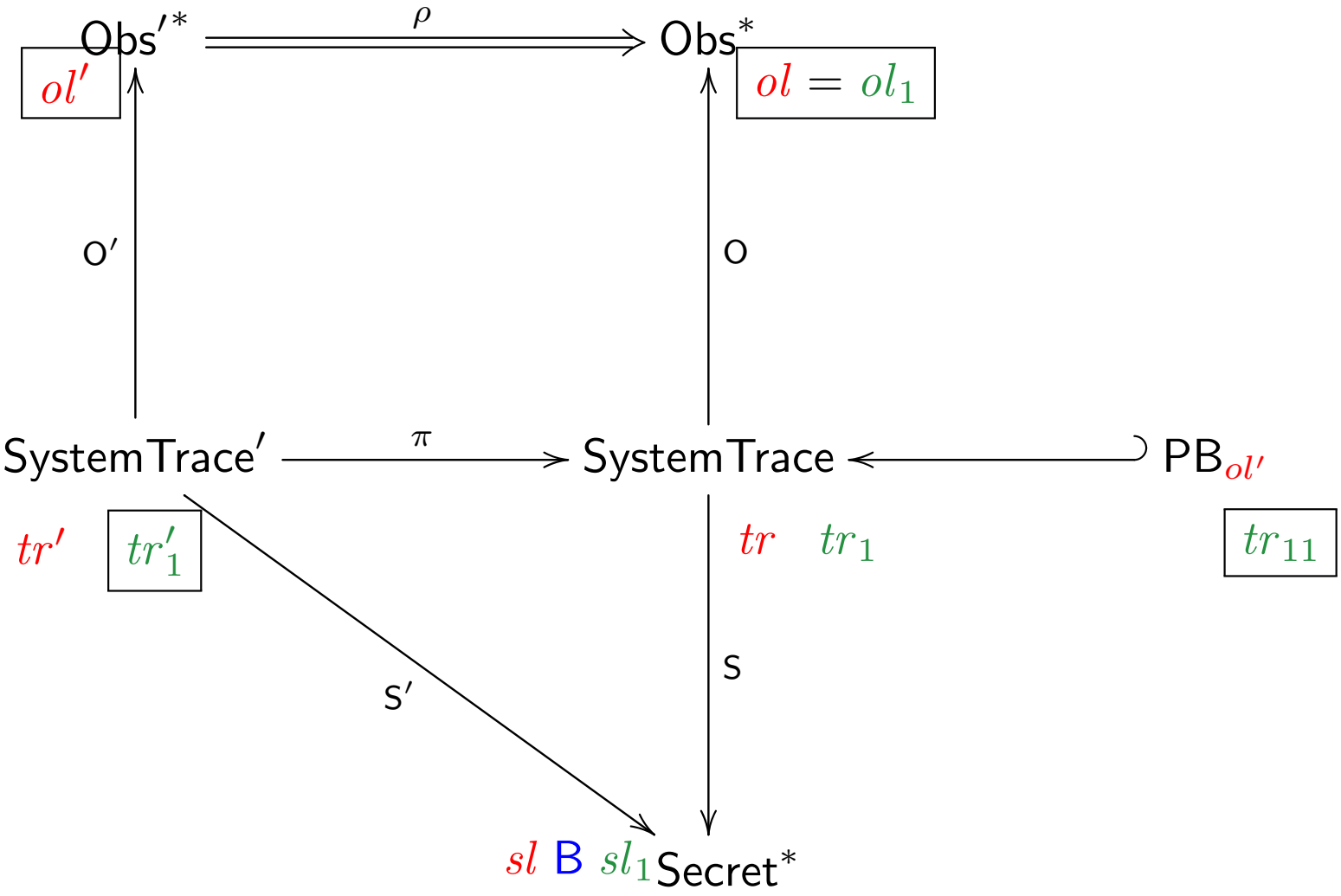
Borrowing BD Security in a Real World



Adjustment property:

$$\forall tr_1, ol'.\ \mathsf{O}\ tr_1 = \rho\ ol' \longrightarrow \exists tr_{11} \in \mathsf{PB}_{ol'}.\ \mathsf{O}\ tr_{11} = \mathsf{O}\ tr_1 \wedge \mathsf{S}\ tr_{11} = \mathsf{S}\ tr_1$$

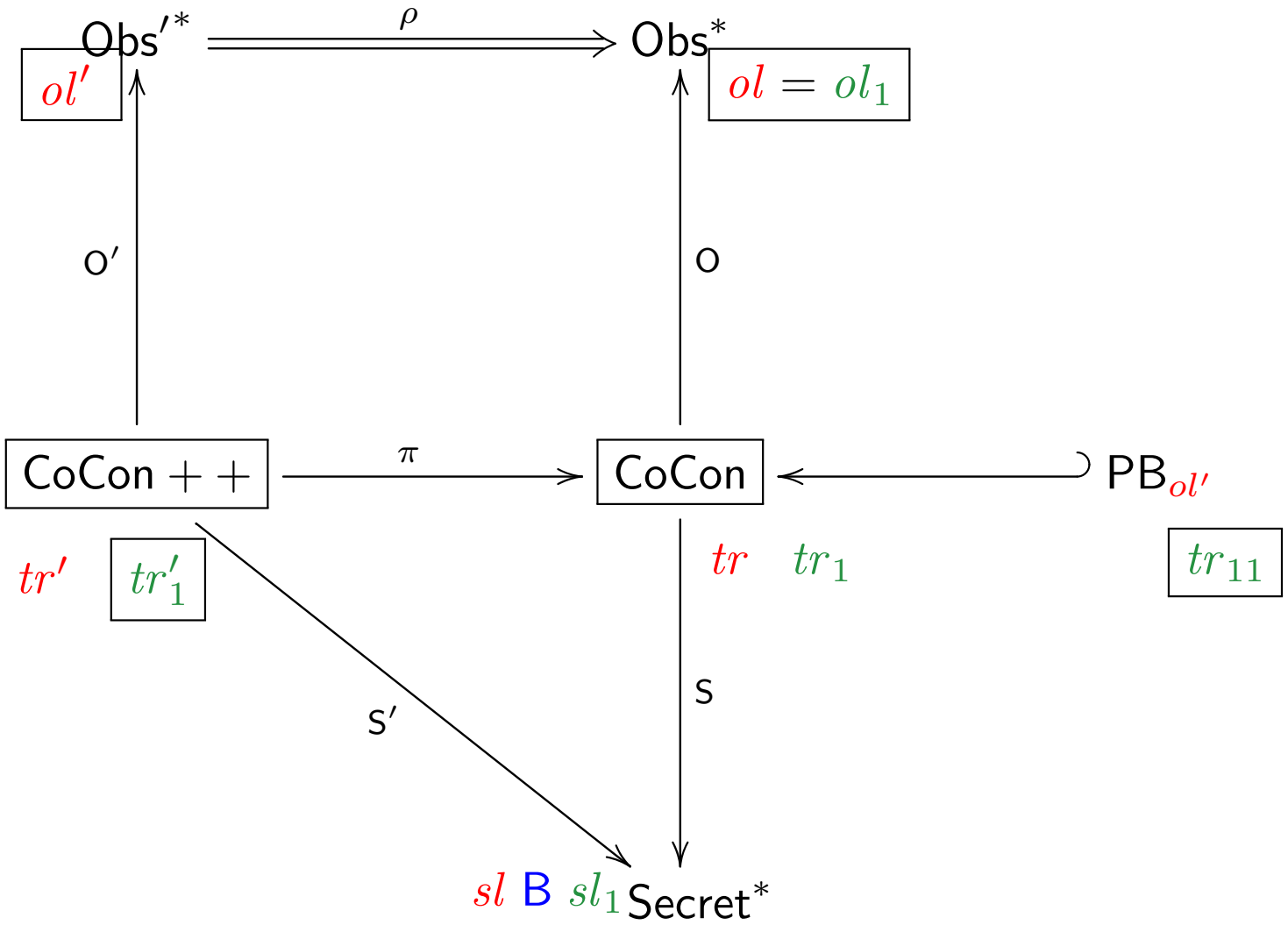
Borrowing BD Security in a Real World



Adjustment property:

$$\forall tr_1,\ ol'.\ \rho\,(\mathsf{O}\ tr_1,\ ol') \longrightarrow \exists tr_{11} \in \mathsf{PB}_{ol'}.\ \mathsf{O}\ tr_{11} = \mathsf{O}\ tr_1 \ \wedge\ \mathsf{S}\ tr_{11} = \mathsf{S}\ tr_1$$

Borrowing BD Security in a Real World



Adjustment property:

$$\forall tr_1,\, ol'.\, \rho\,(\mathsf{O}\ tr_1,\ ol') \longrightarrow \exists tr_{11} \in \mathsf{PB}_{ol'}.\, \mathsf{O}\ tr_{11} = \mathsf{O}\ tr_1 \ \wedge\ \mathsf{S}\ tr_{11} = \mathsf{S}\ tr_1$$

What I Don't Like About CoCon

It's not as much fun as other systems...

Bugs with Happy End

Email subject: [ESOP15] Accepted paper #68

Bugs with Happy End

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.

Bugs with Happy End

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

Ongoing Work – CoSMed

EPSRC grant: Verification or Web-Based Systems (VOWS)

Ongoing Work – CoSMed

EPSRC grant: Verification or Web-Based Systems (VOWS)

Confidentiality-Verified Social Media Platform

Ongoing Work – CoSMed

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

Ongoing Work – CoSMed

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)

Ongoing Work – CoSMed

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

Ongoing Work – CoSMed

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

Ongoing Work – CoSMed

EPSRC grant: Verification or Web-Based Systems (VOWS)

Confidentiality-Verified Social Media Platform

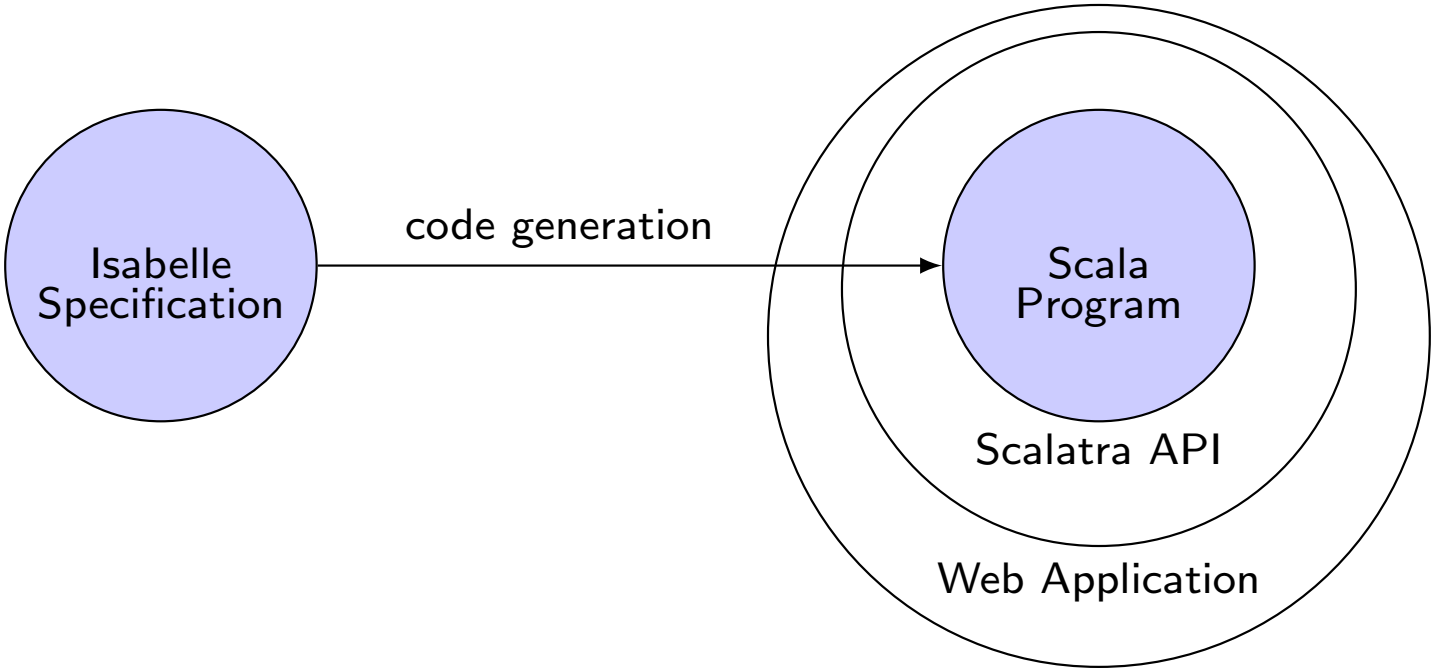
Implemented similarly to CoCon, but includes primitives for communicating with clones of itself — Clones can form a network

Verify each component for confidentiality properties (BD security instance)

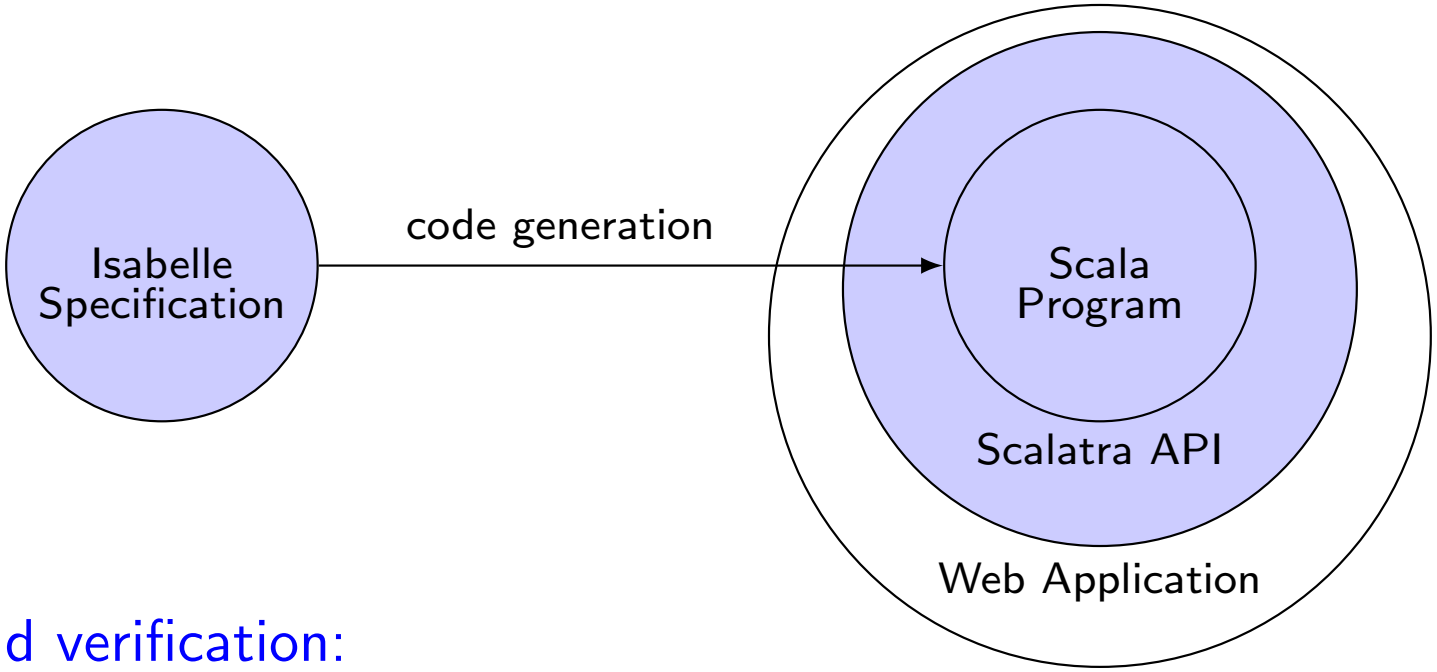
Compositionality results offer guarantees for the whole network

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

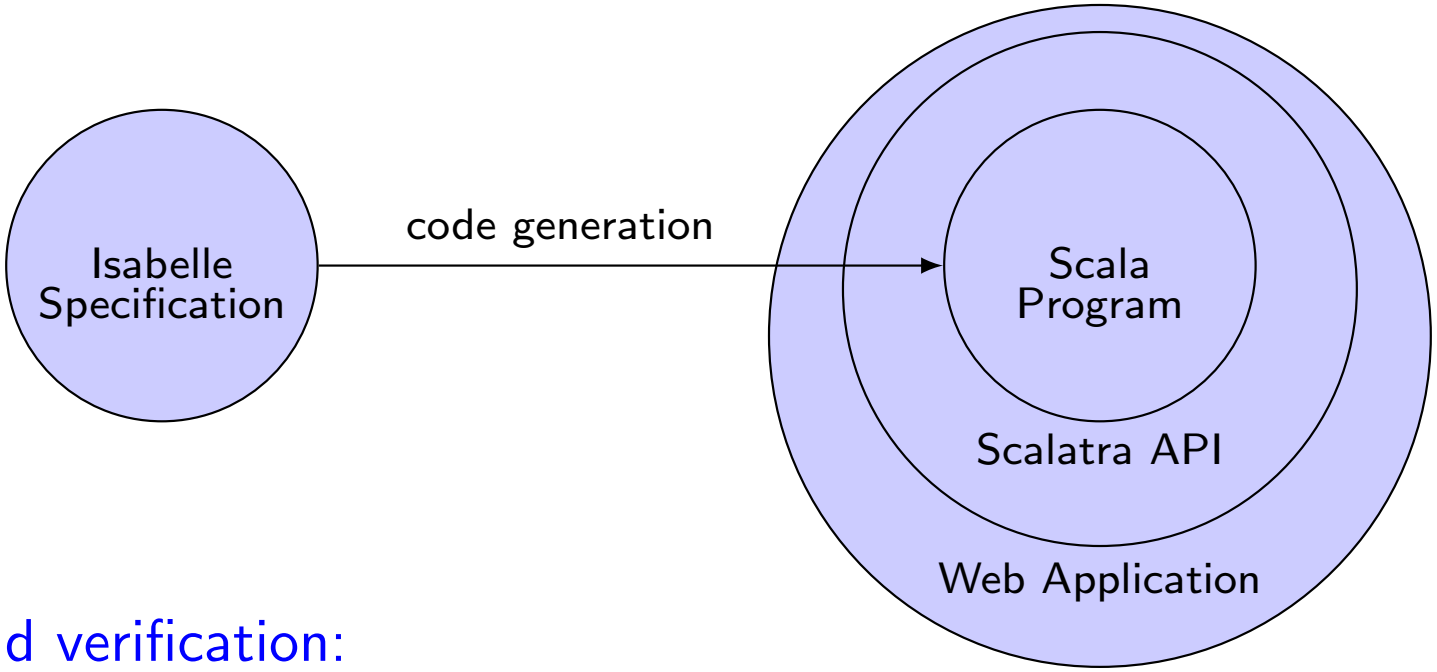


Future Work: End to End Security



End to end verification:
Verify thin API layer

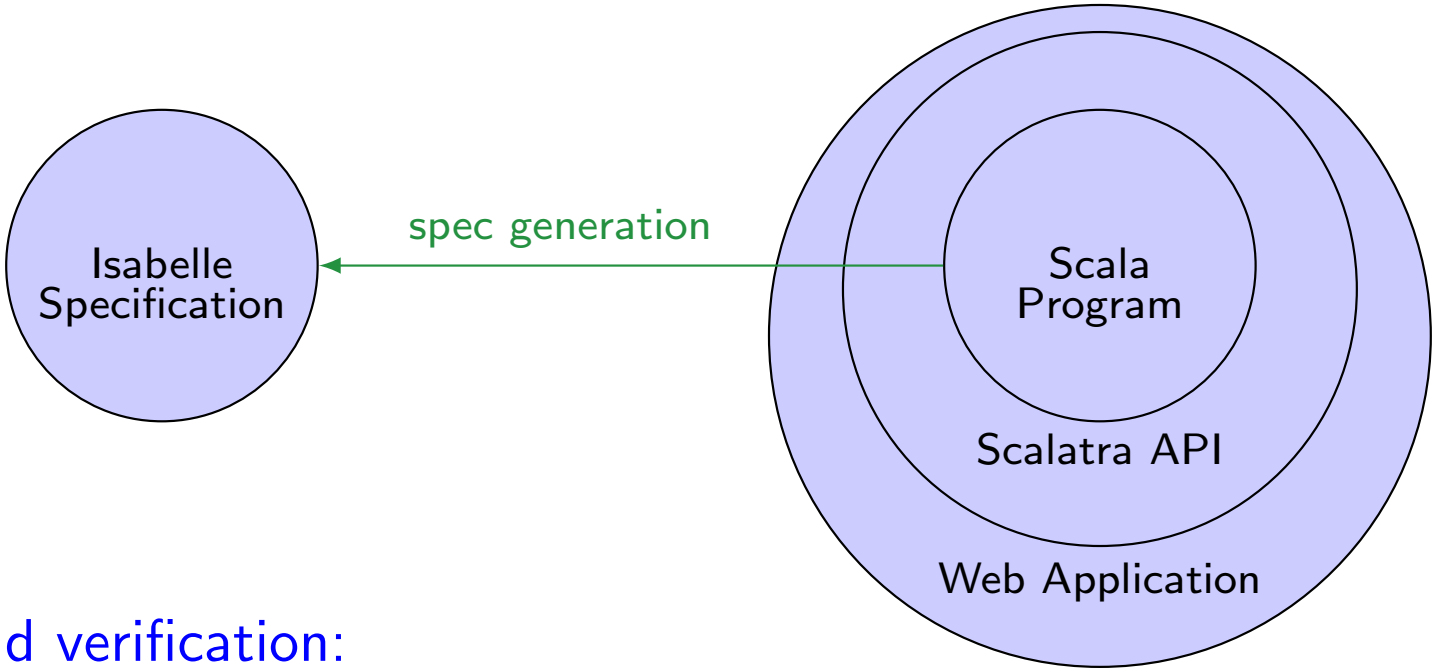
Future Work: End to End Security



End to end verification:

- Verify thin API layer
- Complement with client-side monitoring

Future Work: End to End Security



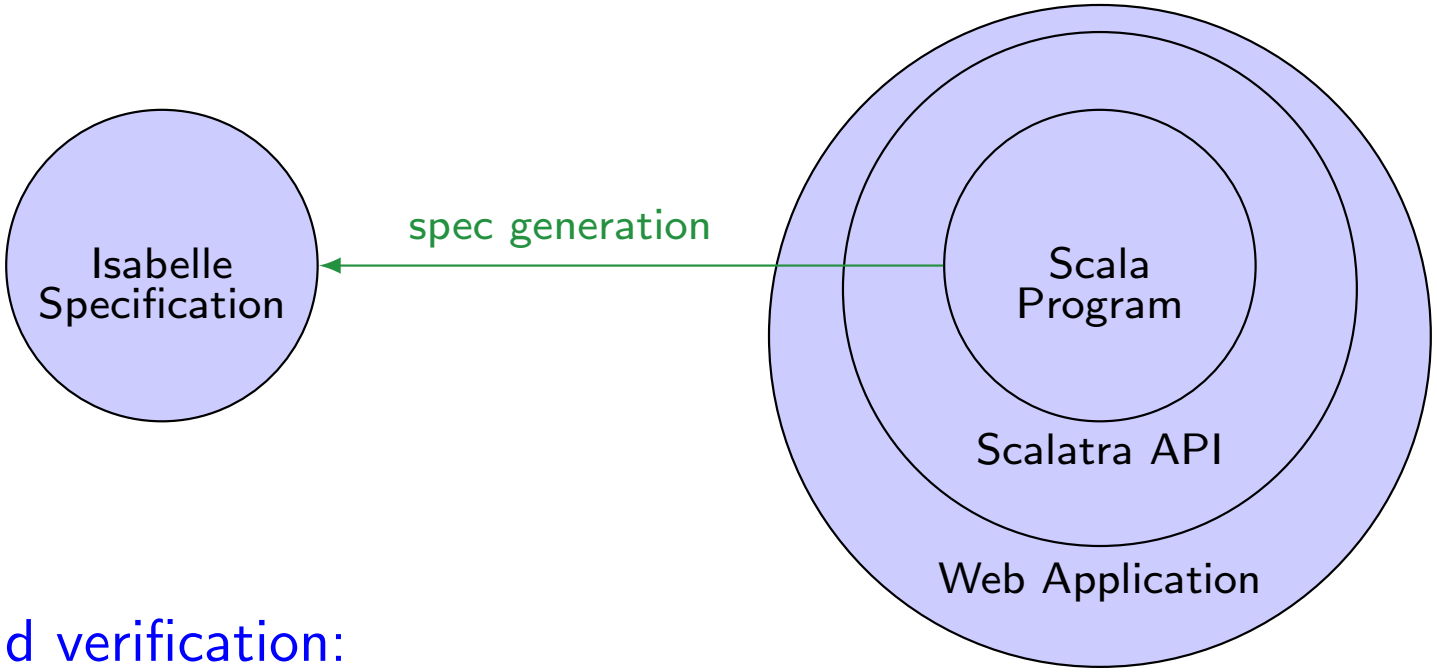
End to end verification:

- 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

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

My Overall Goal

CoCon is starting to replace EasyChair at mainstream conferences

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

My Overall Goal

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

My Overall Goal

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