It's My Privilege: Controlling Downgrading in DC-Labels

Lucas Waye[1], Pablo Buiras[2], Dan King[1], Stephen Chong[1], and Alejandro Russo[2]

[1] Harvard University

[2] Chalmers University of Technology

11th International Workshop on Security and Trust Management

Information Flow Control

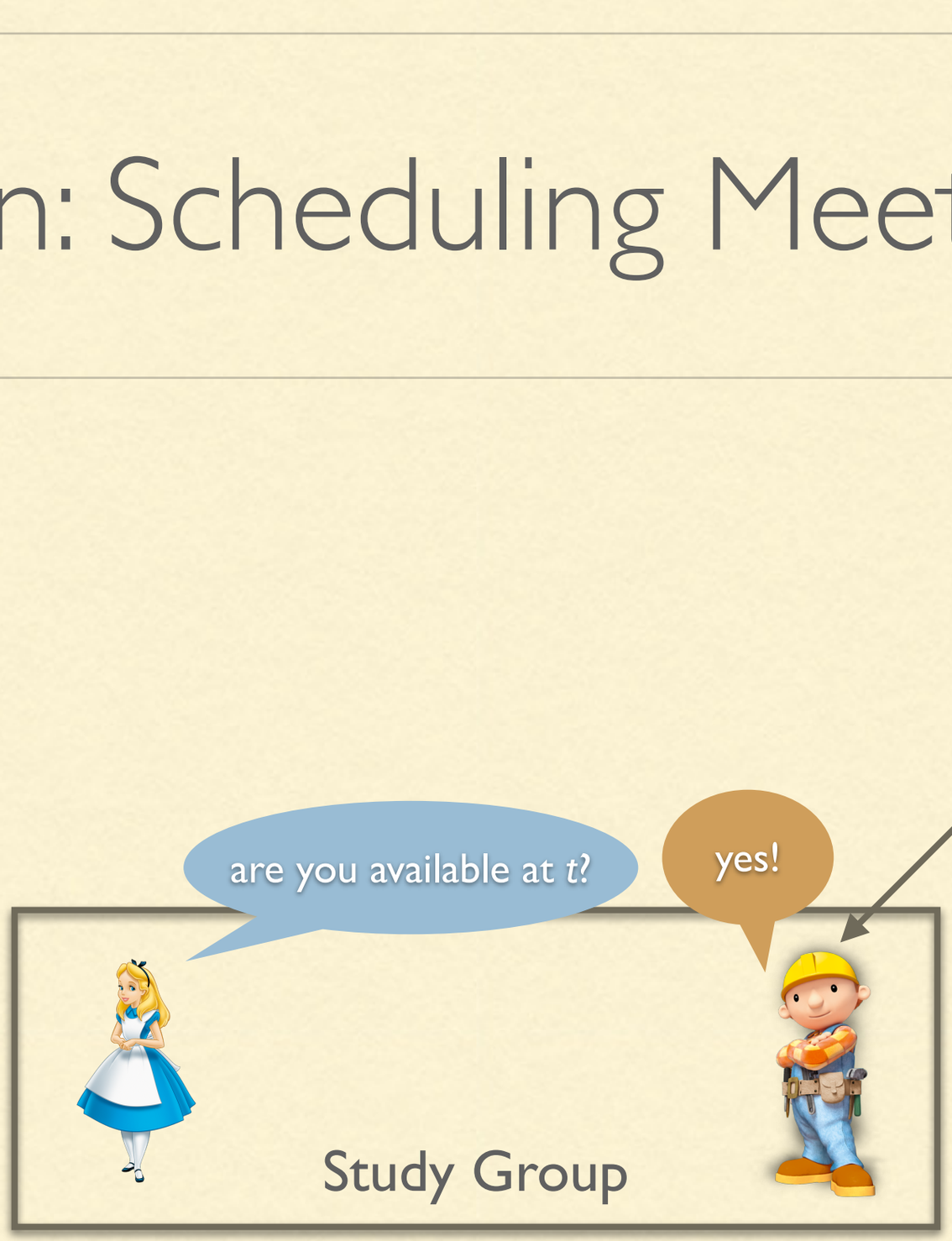
- Can provide strong enforceable security guarantees
  - Noninterference [Goguen and Meseguer 1982]
  - Jif [Myers et al. 2001], LIO [Stefan et al. 2011], and many others
- But programs can **downgrade** information!
  - Breaks noninterference
  - Need semantic security conditions that still hold in the presence of downgrading

This work: reasoning about global security properties in Information Flow Control (IFC) systems with downgrading

Motivation: Scheduling Meetings

Alice's Calendar

Bob's Calendar



group members can discover each other's availability

Scheduling Meetings: constraints

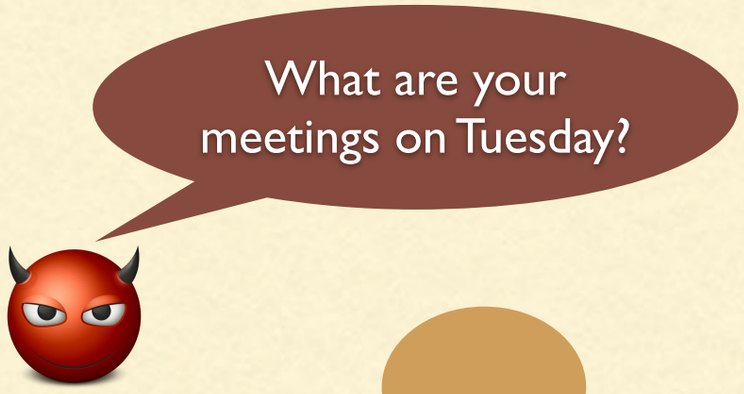
Alice's Calendar

Bob's Calendar

only group members can discover availability

Scheduling Meetings: constraints

Alice's Calendar



Bob's Calendar

meeting details are secret to everyone but the owner

Scheduling Meetings with DC Labels

Disjunction Category (DC) labels provide a way to enforce our security policies

[Stefan et al. 2011]

Scheduling Meetings with DC Labels

Disjunction Category (DC) labels provide a way to enforce our security policies

[Stefan et al. 2011]

Data labeled with information flow policy:

- Confidentiality: who may read the data
- Integrity: who vouches for the data

Scheduling Meetings with DC Labels

Disjunction Category (DC) labels provide a way to enforce our security policies

[Stefan et al. 2011]

Data labeled with information flow policy:

- Confidentiality: who may read the data
- Integrity: who vouches for the data

(Bob, Bob)

Scheduling Meetings with DC Labels

Disjunction Category (DC) labels provide a way to enforce our security policies

[Stefan et al. 2011]

Data labeled with information flow policy:

- Confidentiality: who may read the data
- Integrity: who vouches for the data

(Bob, Bob)

Bob may read the data and vouches for it.

Scheduling Meetings with DC Labels

Disjunction Category (DC) labels provide a way to enforce our security policies

[Stefan et al. 2011]

Data labeled with information flow policy:

- Confidentiality: who may read the data
- Integrity: who vouches for the data

(Bob, Bob)

Bob may read the data and vouches for it.

May compute with labeled data in a *coarse-grained* way

- inside a computational context that has a *floating label*, which reflects the policies of the data that it has used

Scheduling Meetings with DC Labels

Disjunction Category (DC) labels provide a way to enforce our security policies

[Stefan et al. 2011]

Data labeled with information flow policy:

- Confidentiality: who may read the data
- Integrity: who vouches for the data

(Bob, Bob)

Bob may read the data and vouches for it.

May compute with labeled data in a *coarse-grained* way

- inside a computational context that has a *floating label*, which reflects the policies of the data that it has used

(Public, *Player 1*)

Scheduling Meetings with DC Labels

Disjunction Category (DC) labels provide a way to enforce our security policies

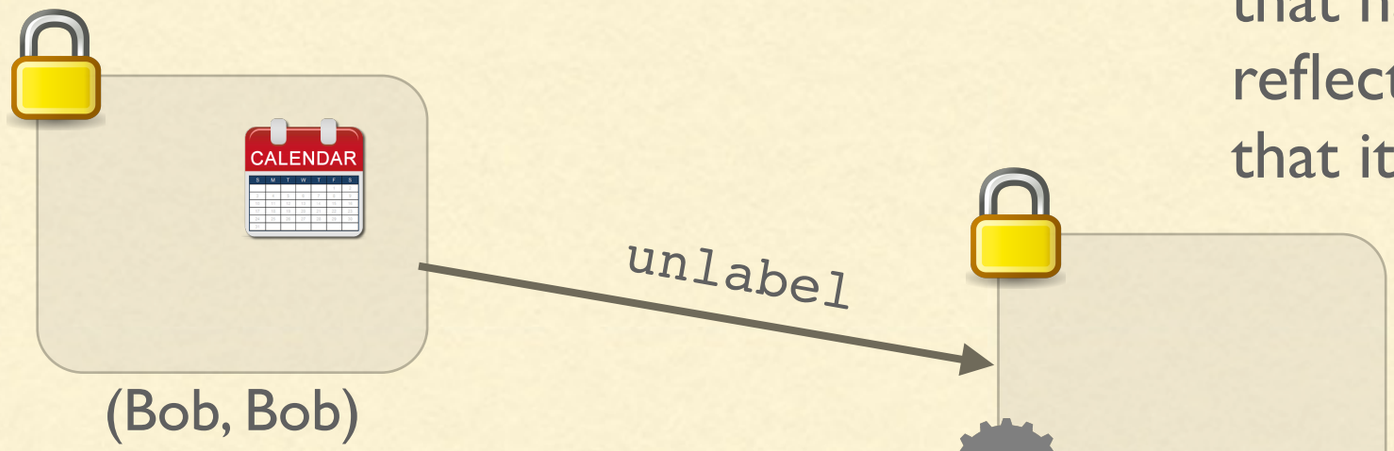
[Stefan et al. 2011]

Data labeled with information flow policy:

- Confidentiality: who may read the data
- Integrity: who vouches for the data

May compute with labeled data in a *coarse-grained* way

- inside a computational context that has a *floating label*, which reflects the policies of the data that it has used



Bob may read the data and vouches for it.

Scheduling Meetings with DC Labels

Disjunction Category (DC) labels provide a way to enforce our security policies

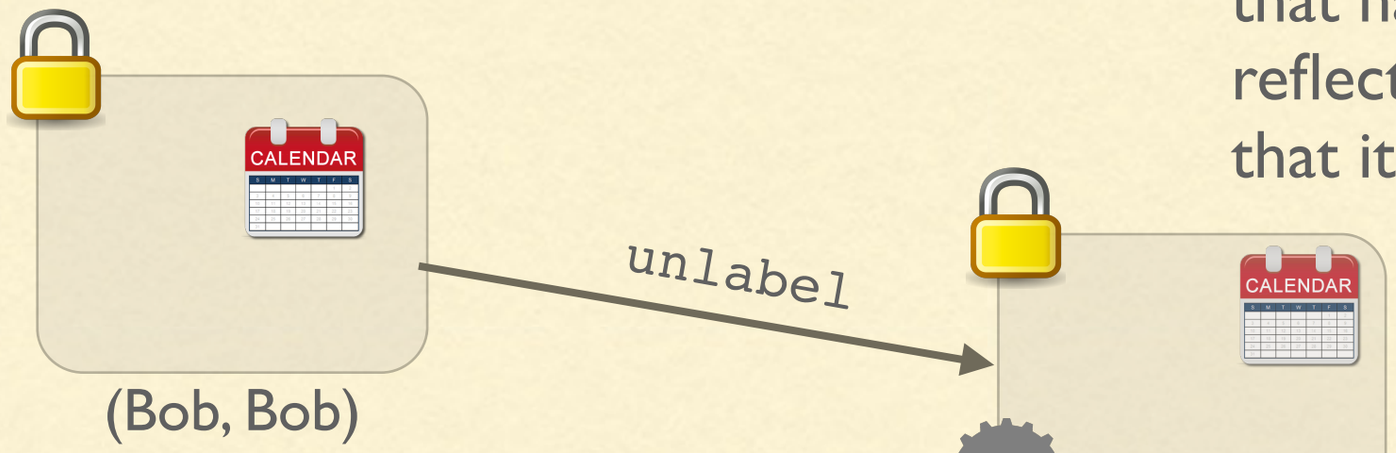
[Stefan et al. 2011]

Data labeled with information flow policy:

- Confidentiality: who may read the data
- Integrity: who vouches for the data

May compute with labeled data in a *coarse-grained* way

- inside a computational context that has a *floating label*, which reflects the policies of the data that it has used



Bob may read the data and vouches for it.

Scheduling Meetings with DC Labels

Disjunction Category (DC) labels provide a way to enforce our security policies

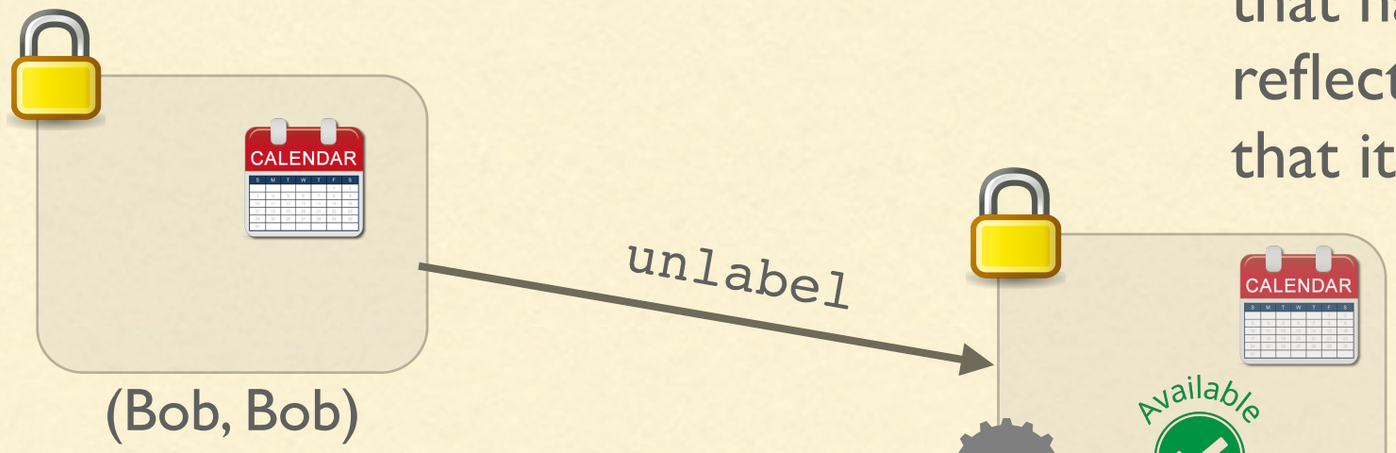
[Stefan et al. 2011]

Data labeled with information flow policy:

- Confidentiality: who may read the data
- Integrity: who vouches for the data

May compute with labeled data in a *coarse-grained* way

- inside a computational context that has a *floating label*, which reflects the policies of the data that it has used



(Bob, Bob)

Bob may read the data and vouches for it.

(~~Public, *Player1*~~)

(Bob, Bob ∨ *Player1*)

Scheduling Meetings with DC Labels

Disjunction Category (DC) labels provide a way to enforce our security policies

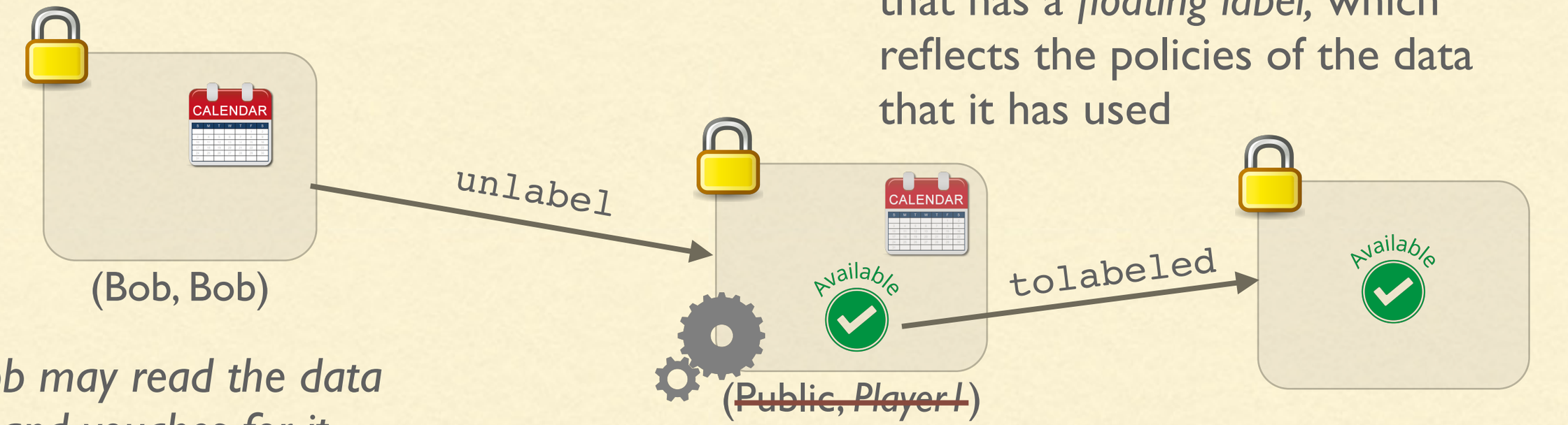
[Stefan et al. 2011]

Data labeled with information flow policy:

- Confidentiality: who may read the data
- Integrity: who vouches for the data

May compute with labeled data in a *coarse-grained* way

- inside a computational context that has a *floating label*, which reflects the policies of the data that it has used



Bob may read the data and vouches for it.

Scheduling Meetings with DC Labels

Disjunction Category (DC) labels provide a way to enforce our security policies

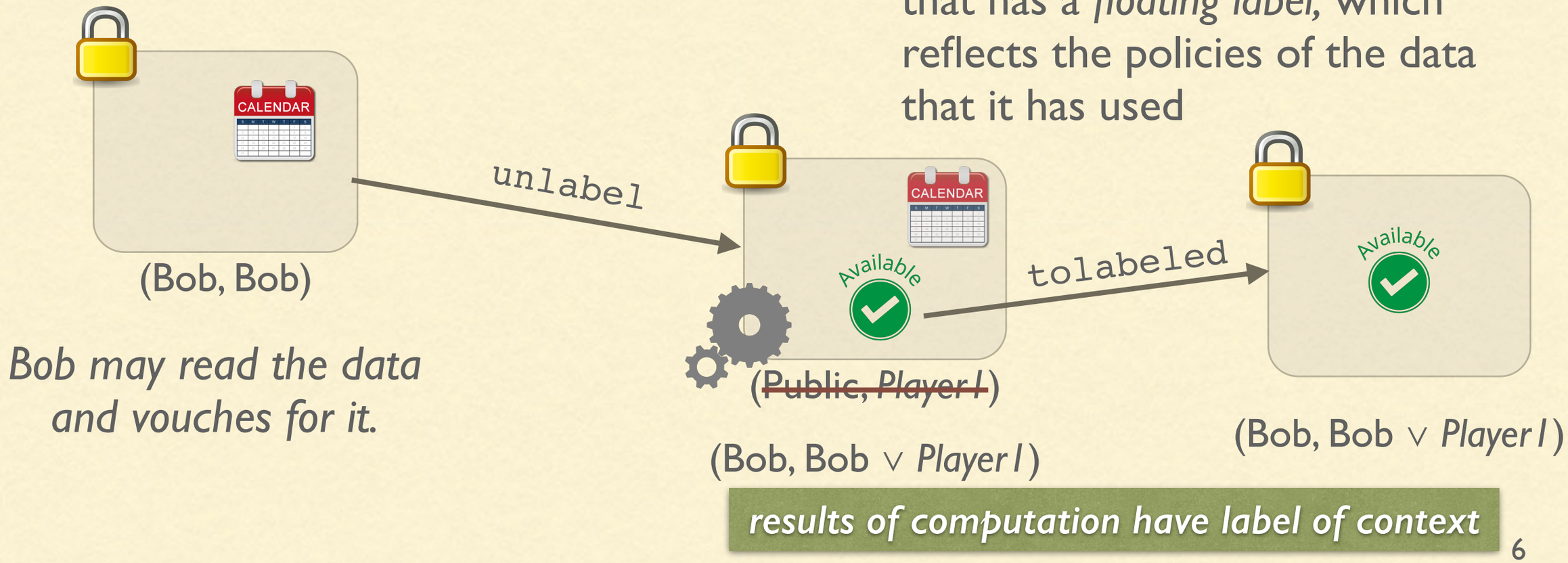
[Stefan et al. 2011]

Data labeled with information flow policy:

- Confidentiality: who may read the data
- Integrity: who vouches for the data

May compute with labeled data in a *coarse-grained* way

- inside a computational context that has a *floating label*, which reflects the policies of the data that it has used



Scheduling Meetings with DC Labels

(Bob, Bob)

Scheduling Meetings with DC Labels

(Bob, Bob)

(Bob, Bob)

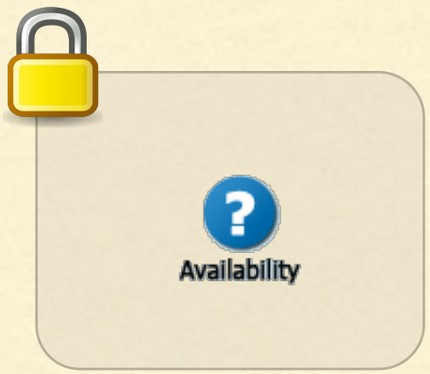
Scheduling Meetings with DC Labels

(Bob, Bob)

(Bob, Bob)

Scheduling Meetings with DC Labels

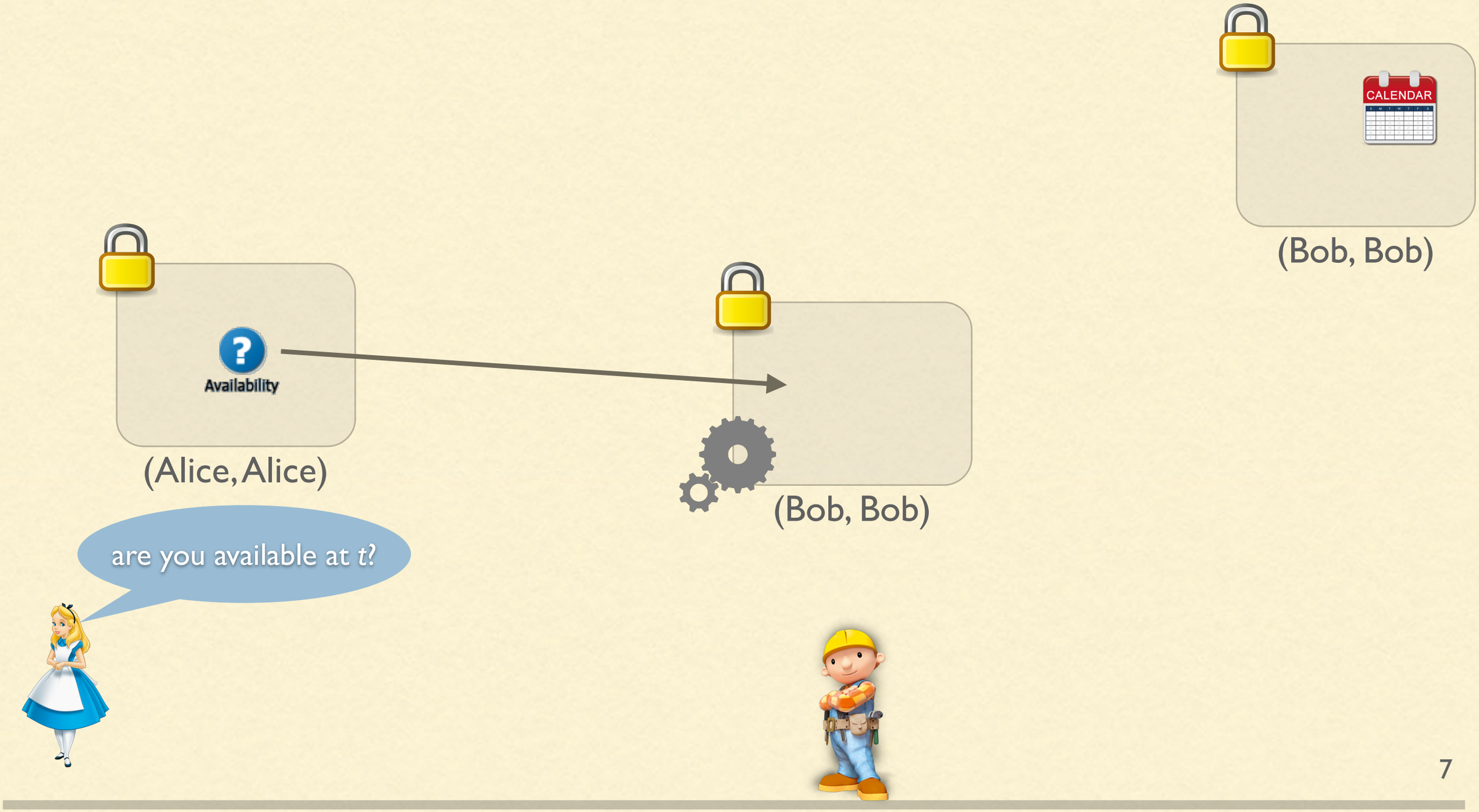
(Bob, Bob)



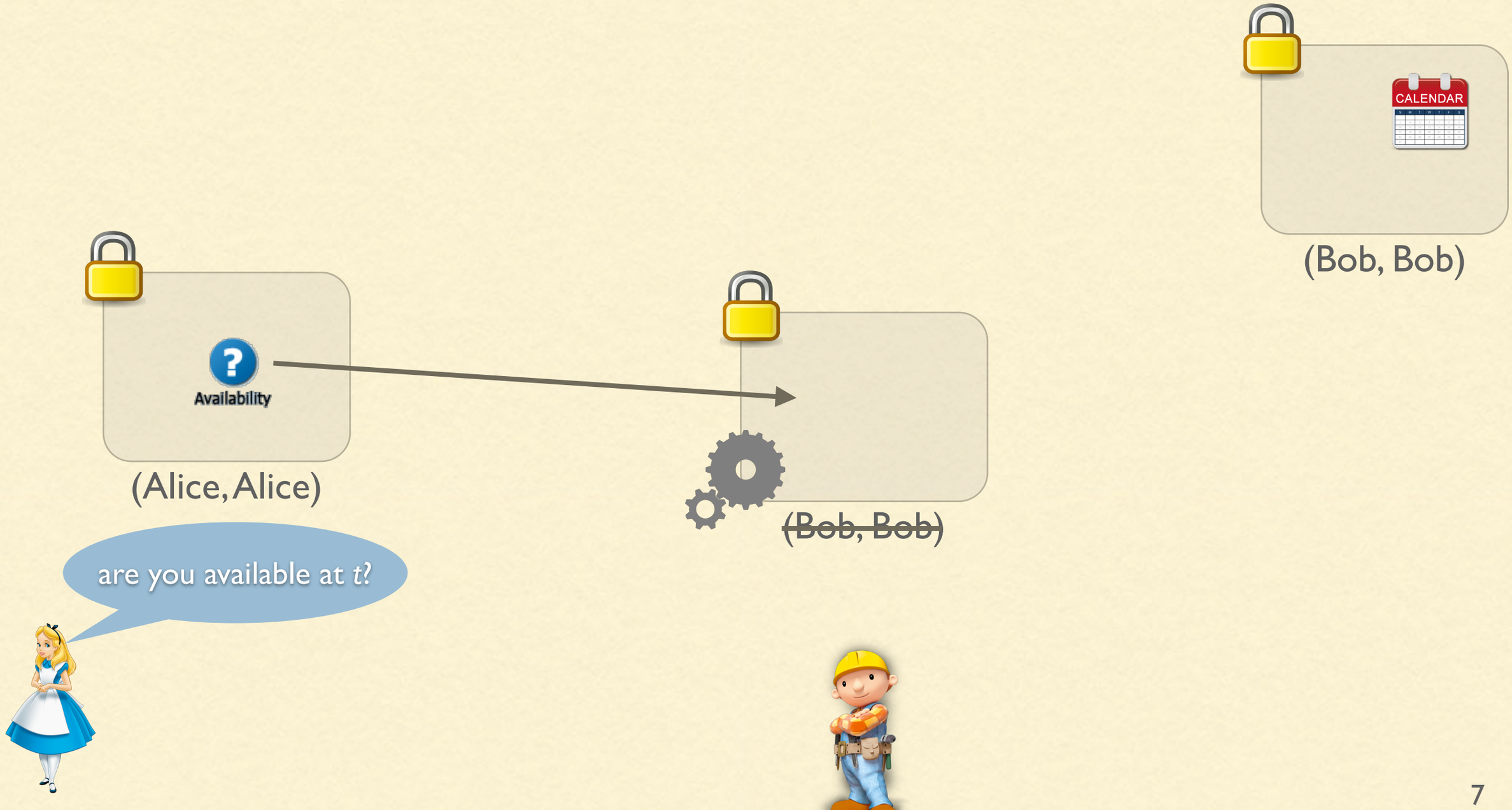
(Alice, Alice)

(Bob, Bob)

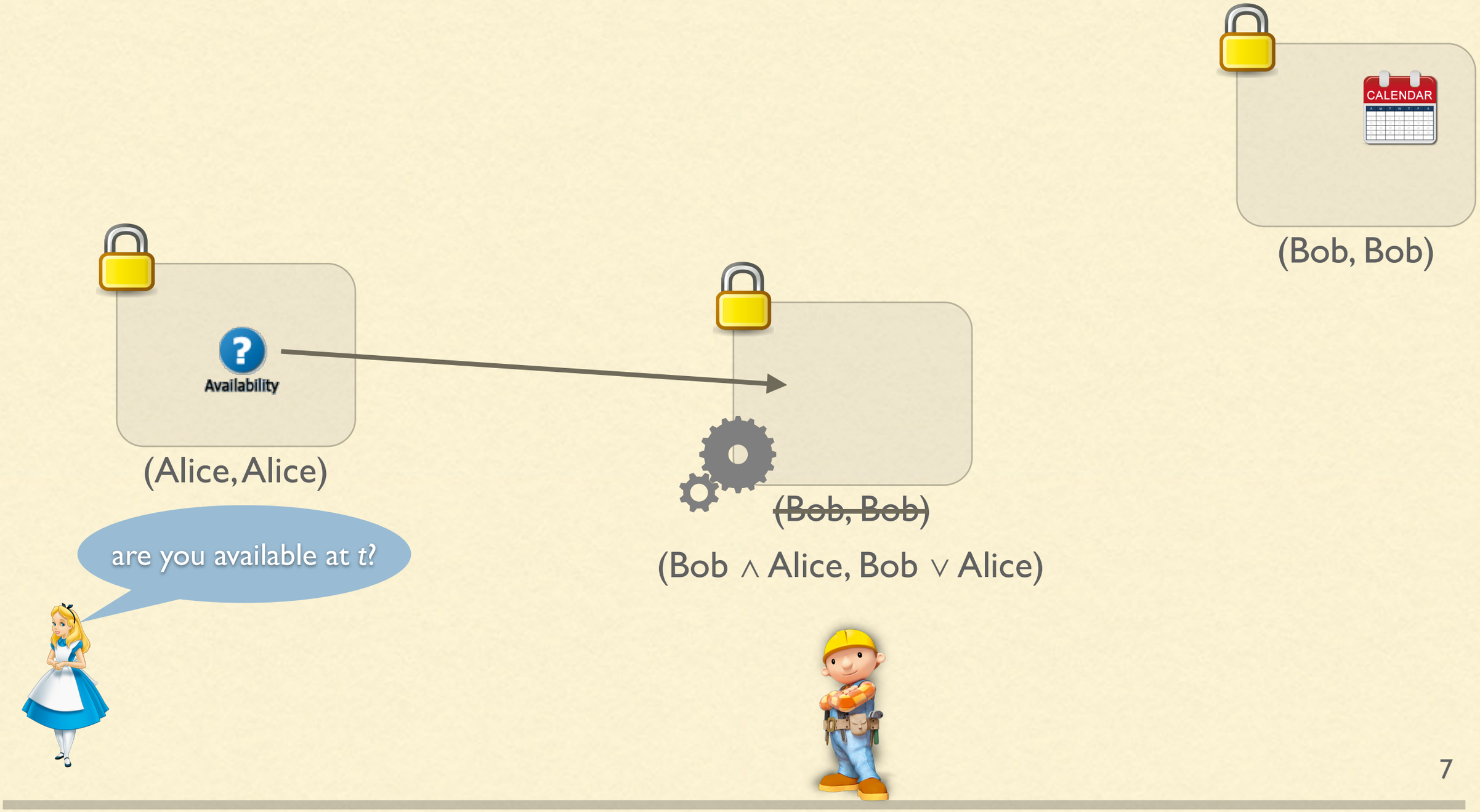
Scheduling Meetings with DC Labels



Scheduling Meetings with DC Labels

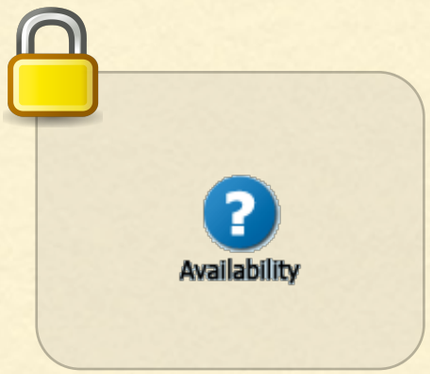


Scheduling Meetings with DC Labels



Scheduling Meetings with DC Labels

(Bob, Bob)

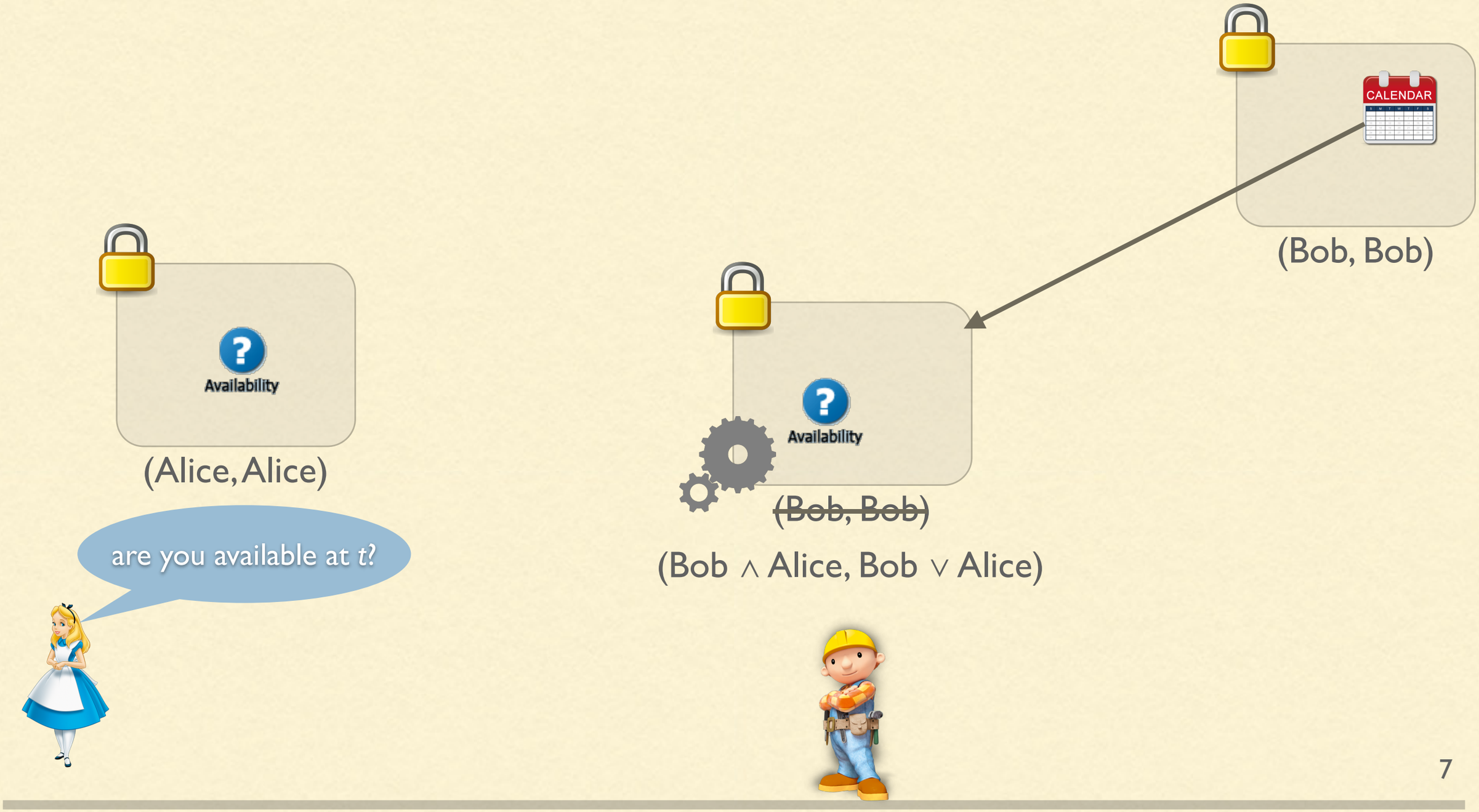


(Alice, Alice)

~~(Bob, Bob)~~

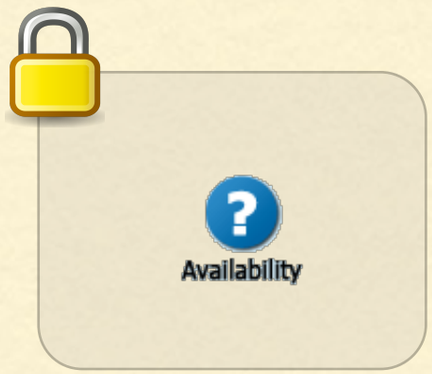
(Bob ∧ Alice, Bob ∨ Alice)

Scheduling Meetings with DC Labels

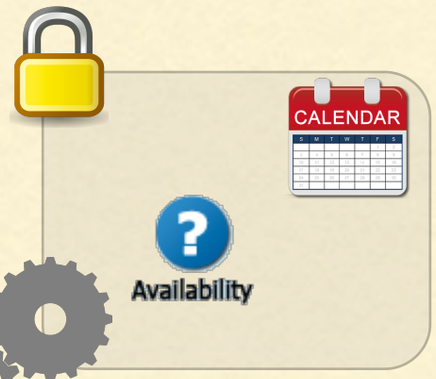


Scheduling Meetings with DC Labels

(Bob, Bob)



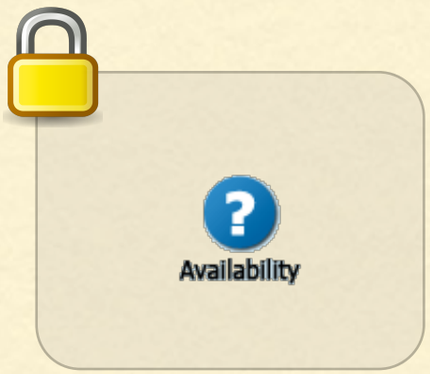
(Alice, Alice)



~~(Bob, Bob)~~

(Bob ∧ Alice, Bob ∨ Alice)

Scheduling Meetings with DC Labels



(Alice, Alice)

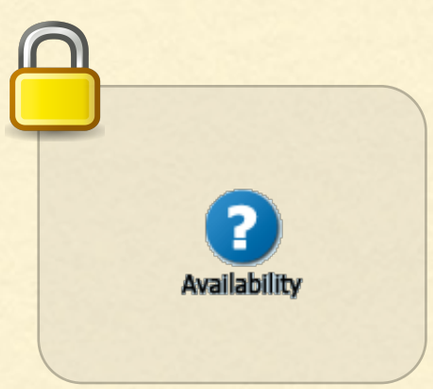
(Bob, Bob)

~~(Bob, Bob)~~

(Bob ∧ Alice, Bob ∨ Alice)

Scheduling Meetings with DC Labels

(Bob, Bob)



(Alice, Alice)

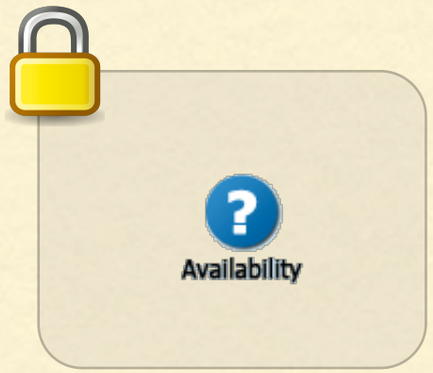
~~(Bob, Bob)~~

(Bob ∧ Alice, Bob ∨ Alice)

(Bob ∧ Alice, Bob ∨ Alice)

Scheduling Meetings with DC Labels

(Bob, Bob)



(Alice, Alice)

~~(Bob, Bob)~~

(Bob ∧ Alice, Bob ∨ Alice)

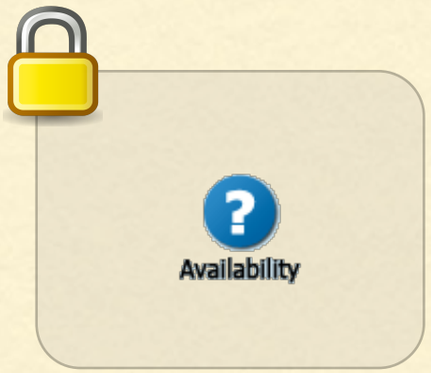
(Bob ∧ Alice, Bob ∨ Alice)

unreadable by Alice

Scheduling Meetings with DC Labels

Privilege: capability-like runtime value that represents a principal's authority to downgrade information

(Bob, Bob)



(Alice, Alice)

~~(Bob, Bob)~~

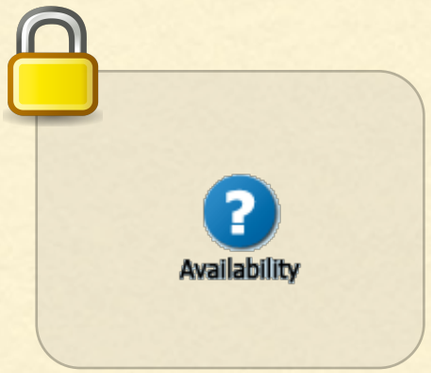
(Bob ∧ Alice, Bob ∨ Alice)

(Bob ∧ Alice, Bob ∨ Alice)

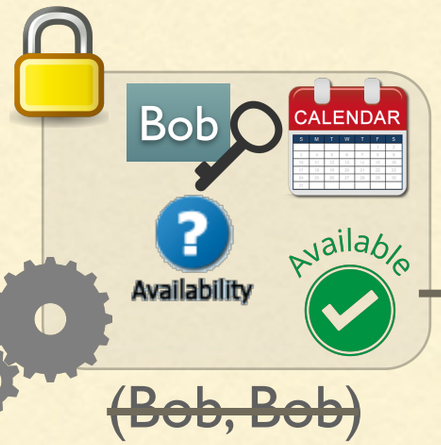
Scheduling Meetings with DC Labels

Privilege: capability-like runtime value that represents a principal's authority to downgrade information

(Bob, Bob)



(Alice, Alice)



~~(Bob, Bob)~~

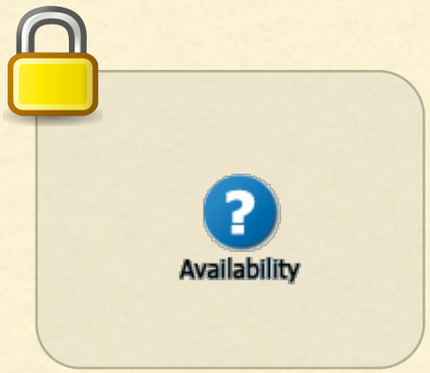
(Bob ∧ Alice, Bob ∨ Alice)

(Bob ∧ Alice, Bob ∨ Alice)

Scheduling Meetings with DC Labels

Privilege: capability-like runtime value that represents a principal's authority to downgrade information

(Bob, Bob)



(Alice, Alice)

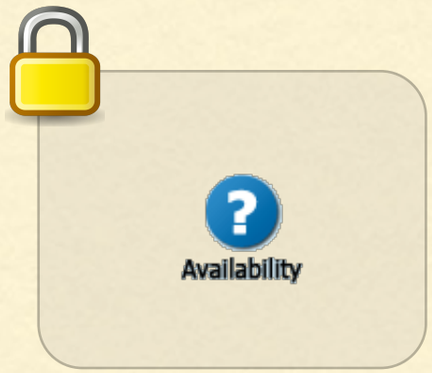
~~(Bob, Bob)~~

(Bob ∧ Alice, Bob ∨ Alice)

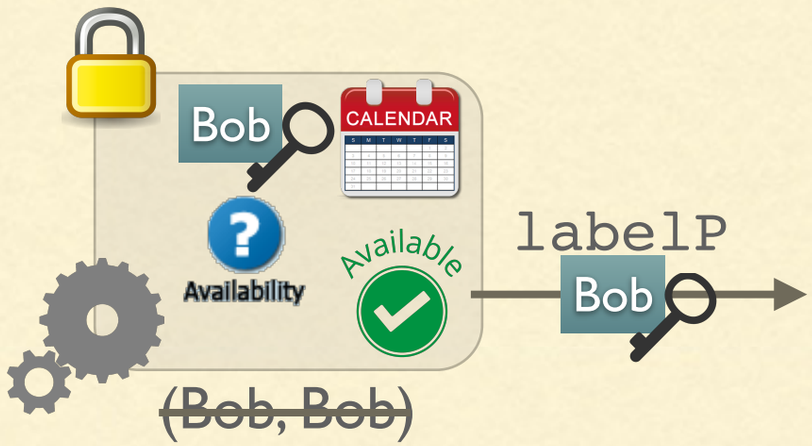
Scheduling Meetings with DC Labels

Privilege: capability-like runtime value that represents a principal's authority to downgrade information

(Bob, Bob)



(Alice, Alice)



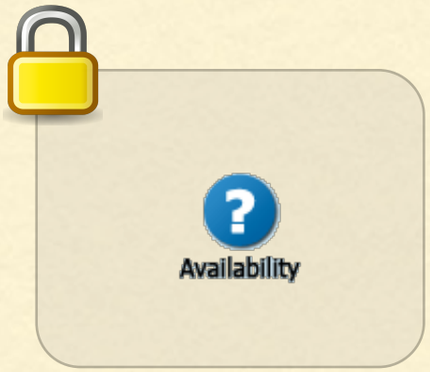
~~(Bob, Bob)~~

(Bob ∧ Alice, Bob ∨ Alice)

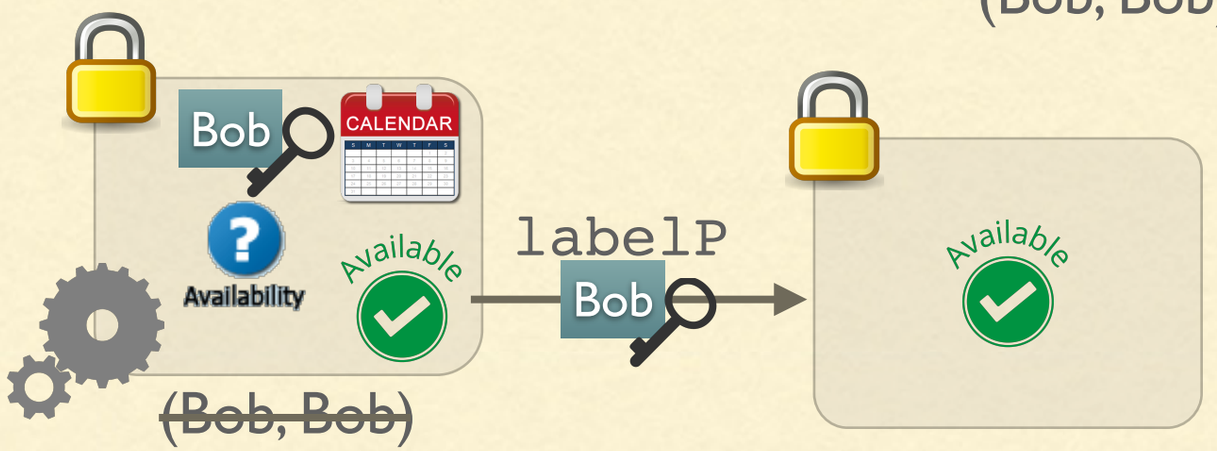
Scheduling Meetings with DC Labels

Privilege: capability-like runtime value that represents a principal's authority to downgrade information

(Bob, Bob)



(Alice, Alice)



~~(Bob, Bob)~~

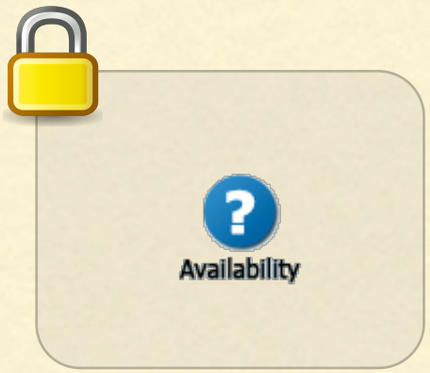
(Bob ∧ Alice, Bob ∨ Alice)

(Alice, Bob ∨ Alice)

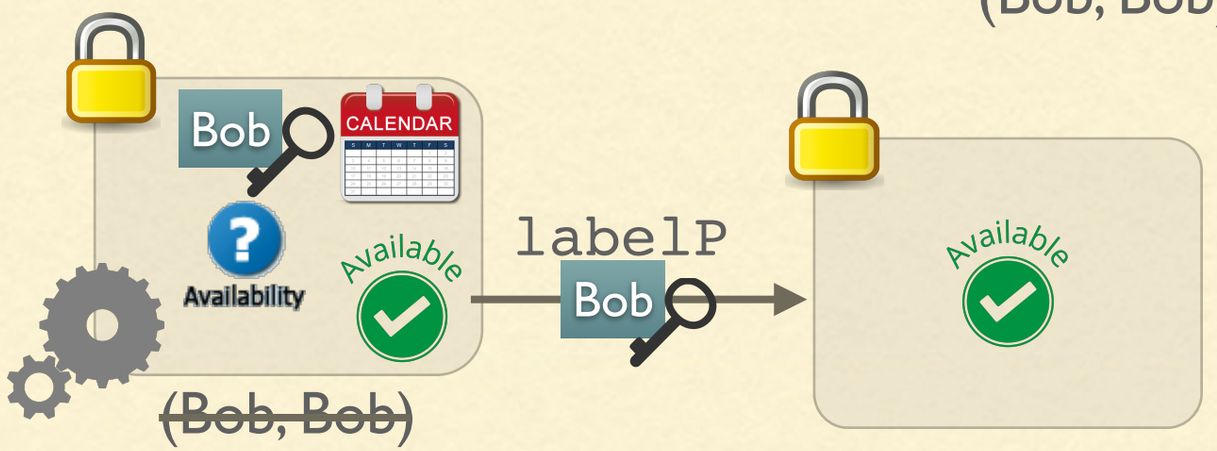
Scheduling Meetings with DC Labels

Privilege: capability-like runtime value that represents a principal's authority to downgrade information

(Bob, Bob)



(Alice, Alice)

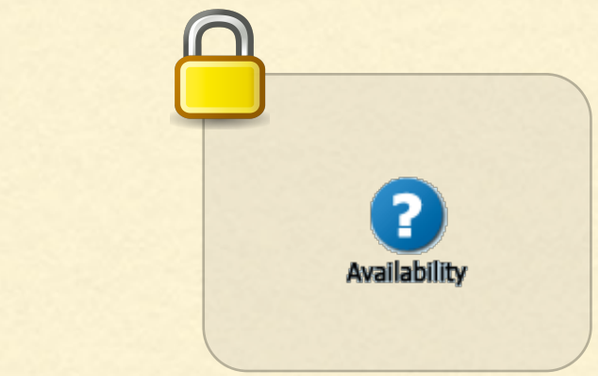


~~(Bob, Bob)~~

(Bob ∧ Alice, Bob ∨ Alice)

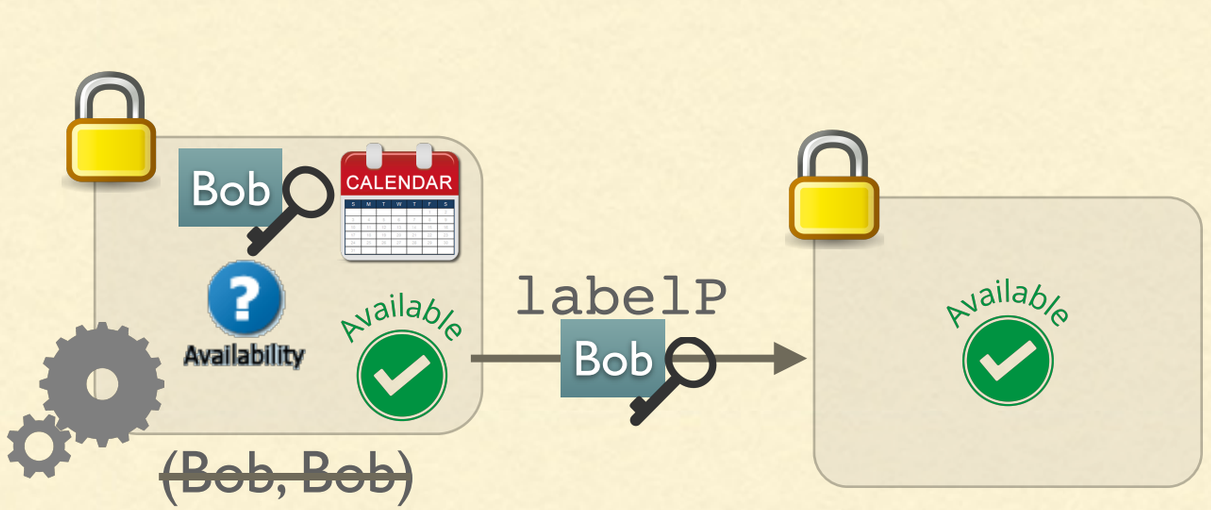
(Alice, Bob ∨ Alice)

Dangerous Downgrading



(Eve, Eve)

(not in Bob's study group)

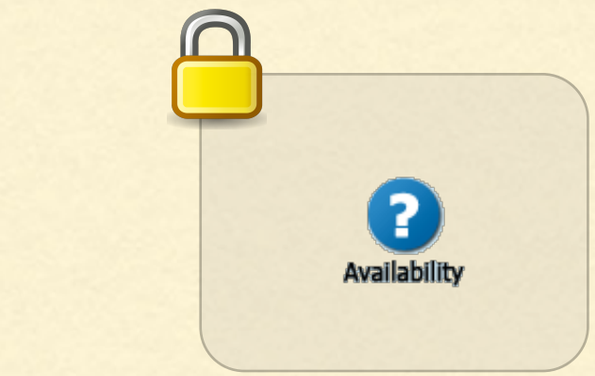


~~(Bob, Bob)~~

(Bob ∧ Eve, Bob ∨ Eve)

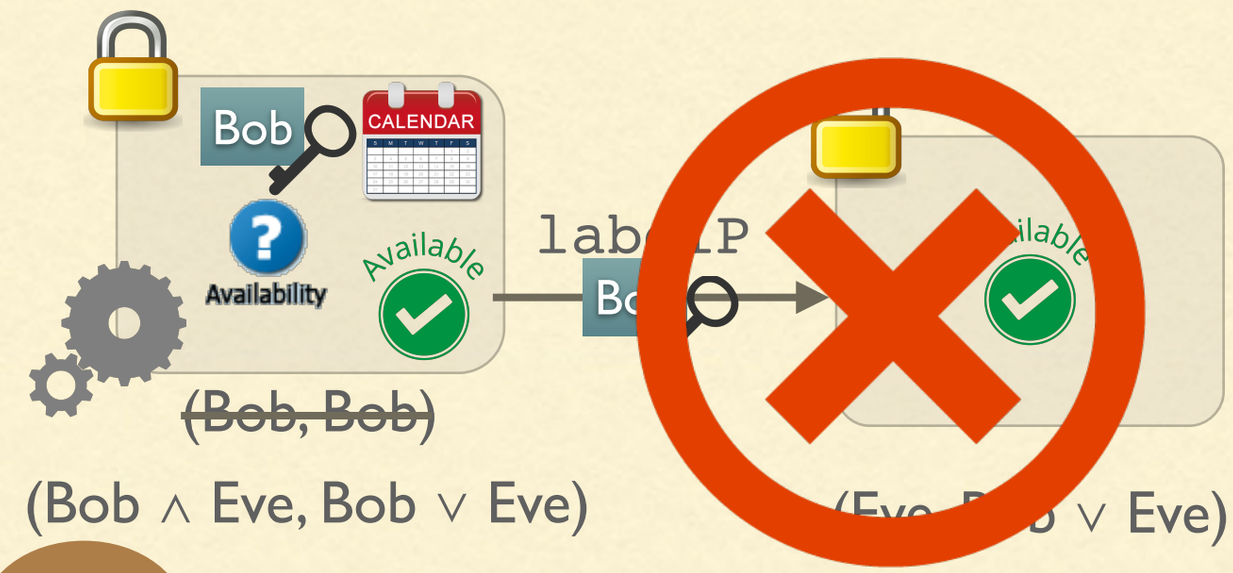
(Eve, Bob ∨ Eve)

Dangerous Downgrading



(Eve, Eve)

(not in Bob's study group)



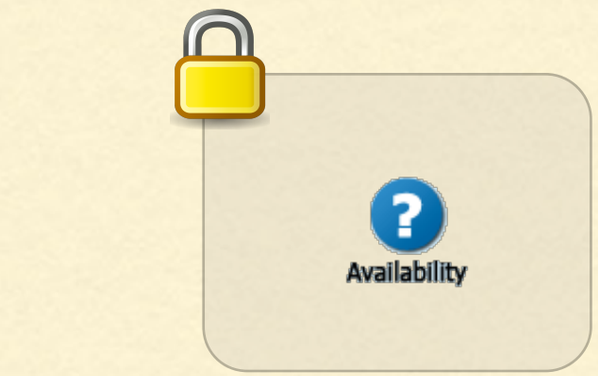
~~(Bob, Bob)~~

(Bob ∧ Eve, Bob ∨ Eve)

(Eve, Bob ∨ Eve)

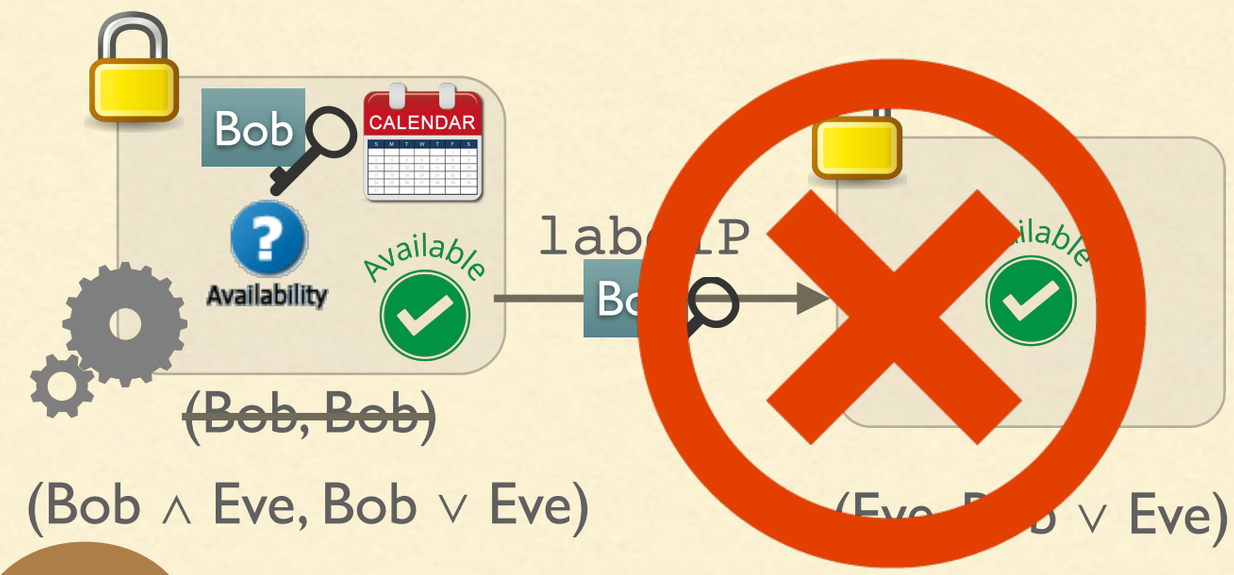
Dangerous Downgrading

Correctness of some security properties requires reasoning about the use of dangerous operations in the application



(Eve, Eve)

(not in Bob's study group)



~~(Bob, Bob)~~

(Bob ∧ Eve, Bob ∨ Eve)

(Eve, Bob ∨ Eve)

This work

Problem: does app use privileges correctly? May require examining all app code!

- even minor mistakes can violate security (see paper)

Our solution: restricted privileges

- declarative restrictions on privileges
- enable local reasoning about global security

We present a semantic characterization of how restricted privileges limit downgrading and evaluate their usefulness in a case study

Restricted Privileges

- **Bounded Privileges**
  - Bounds where in the security lattice downgrading can occur

Restricted Privileges

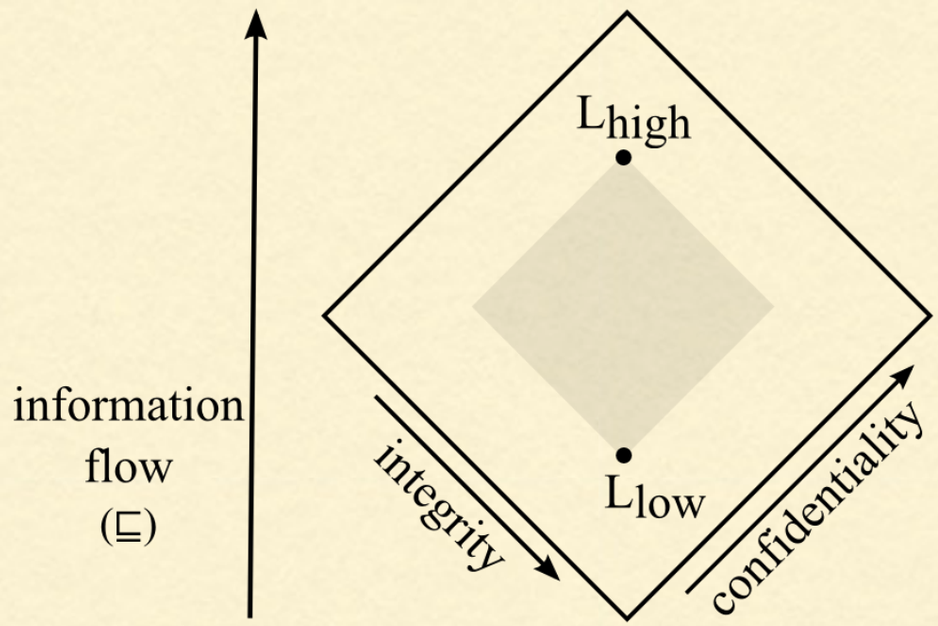
- **Bounded Privileges**
  - Bounds where in the security lattice downgrading can occur
- **Robust Privileges**
  - Restricts downgrading of information so that those influencing the declassification do not benefit from it

Bounded Privileges

Bounds *where* [Sabelfeld and Sands 2005] in the security lattice a downgrading may occur.

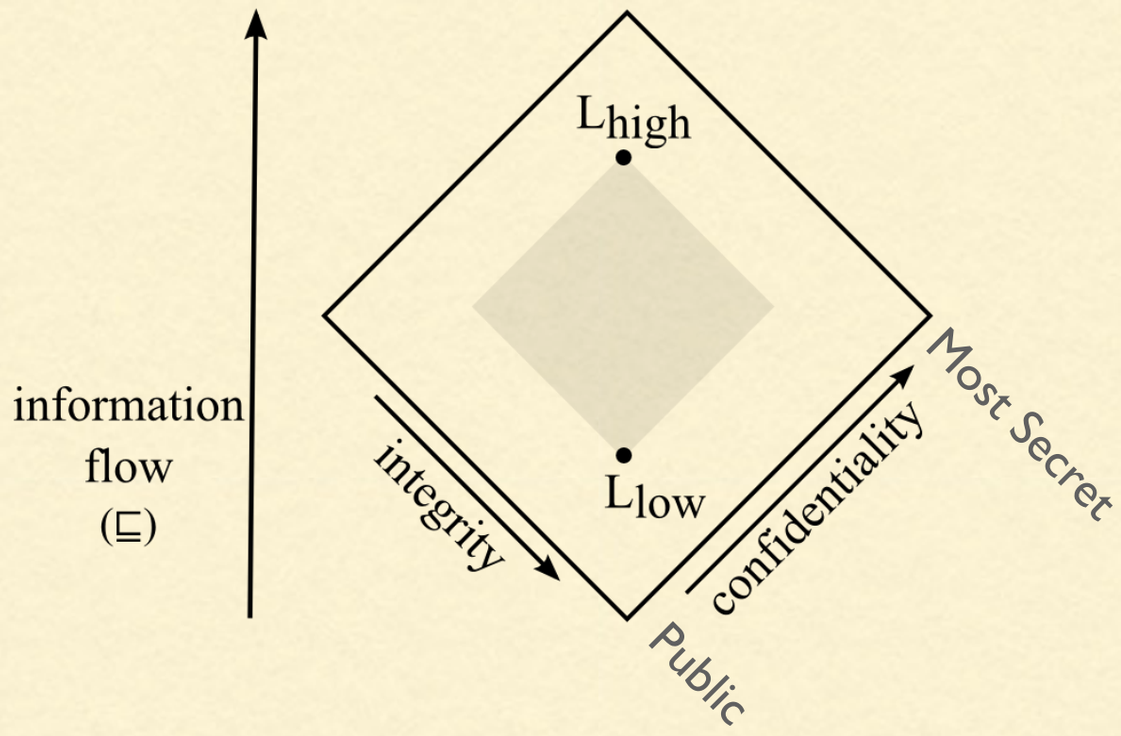
Bounded Privileges

Bounds *where* [Sabelfeld and Sands 2005] in the security lattice a downgrading may occur.



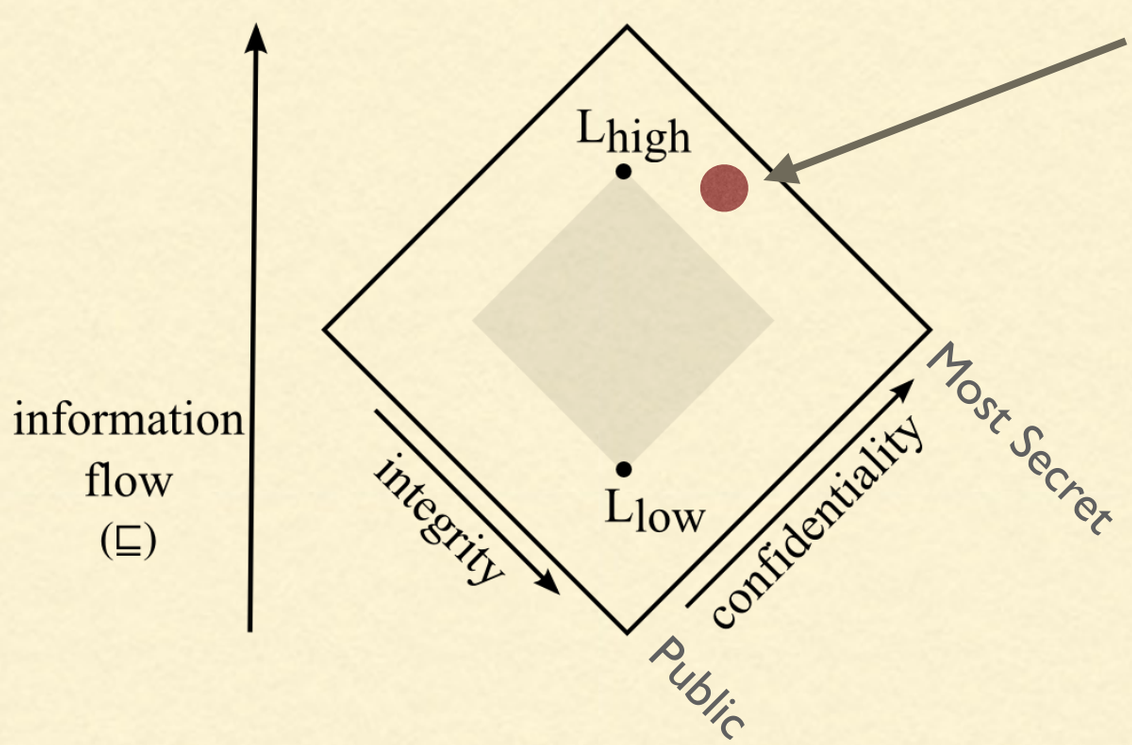
Bounded Privileges

Bounds *where* [Sabelfeld and Sands 2005] in the security lattice a downgrading may occur.



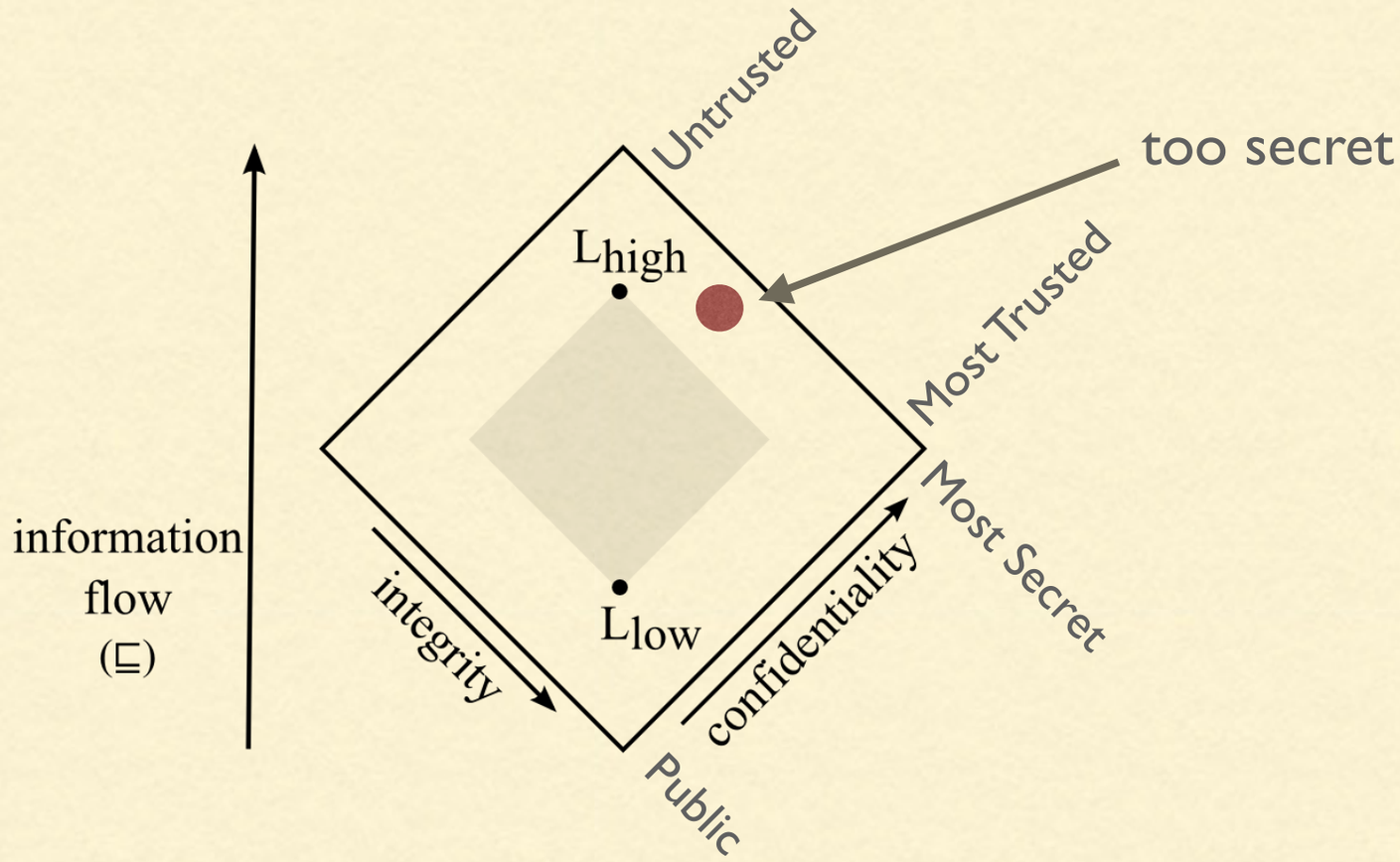
Bounded Privileges

Bounds *where* [Sabelfeld and Sands 2005] in the security lattice a downgrading may occur.



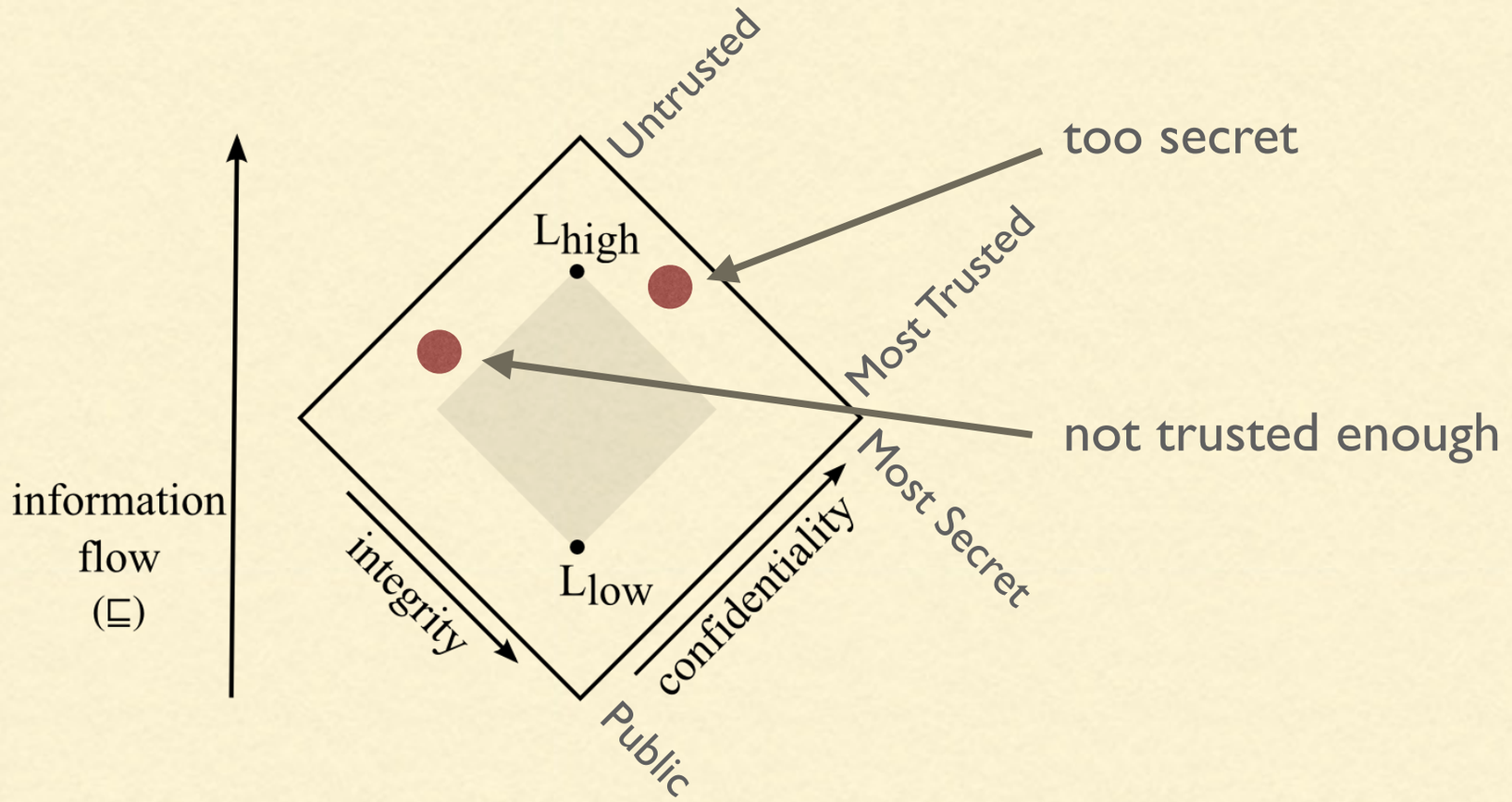
Bounded Privileges

Bounds *where* [Sabelfeld and Sands 2005] in the security lattice a downgrading may occur.



Bounded Privileges

Bounds *where* [Sabelfeld and Sands 2005] in the security lattice a downgrading may occur.

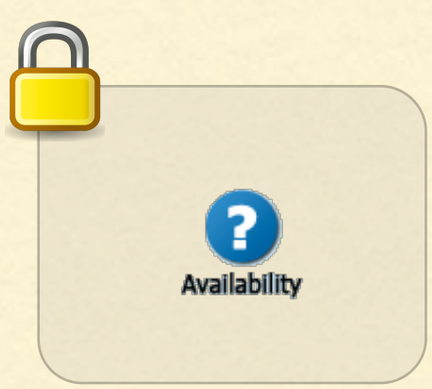


Bounded Privileges

Upper bound: (*Most Secret*, *Untrusted*)

Lower bound: (Alice, *Most Trusted*)

(Bob, Bob)

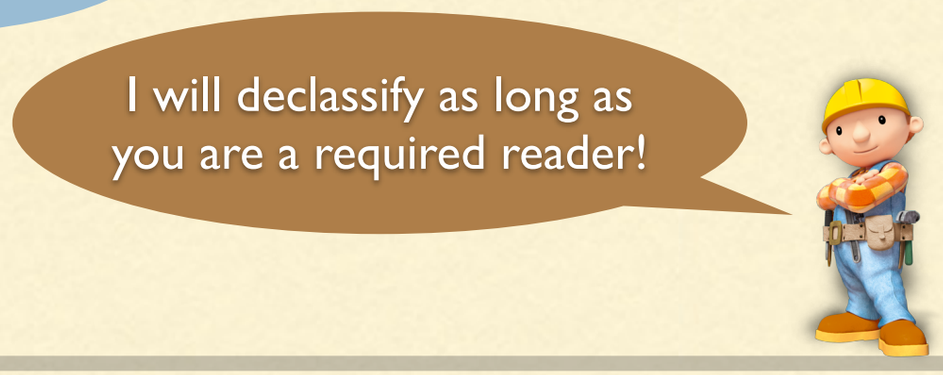


(Alice, Alice)

(Bob ∧ Alice, Bob ∨ Alice)

labelP

(Alice, Bob ∨ Alice)



Robust Privileges

- Based on **robust declassification** [Zdancewic and Myers 2001]
- Restricts *who* [Sabelfeld and Sands 2005] can influence a declassification
- **Intuition**: those who benefit from a declassification had no influence on the data or the decision to declassify.

Robust Privileges

- Based on **robust declassification** [Zdancewic and Myers 2001]
- Restricts *who* [Sabelfeld and Sands 2005] can influence a declassification
- **Intuition**: those who benefit from a declassification had no influence on the data or the decision to declassify.

A formula is robust, *if:*

For all boolean formulas of principals `A`, *if*

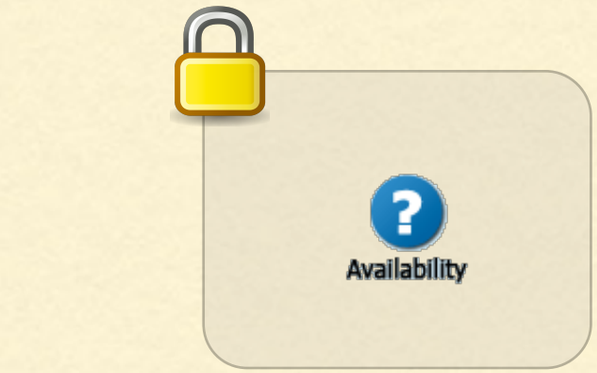
`A` can read the data but could not before the declassification

then,

`A` was not (even partially) responsible for the data, and

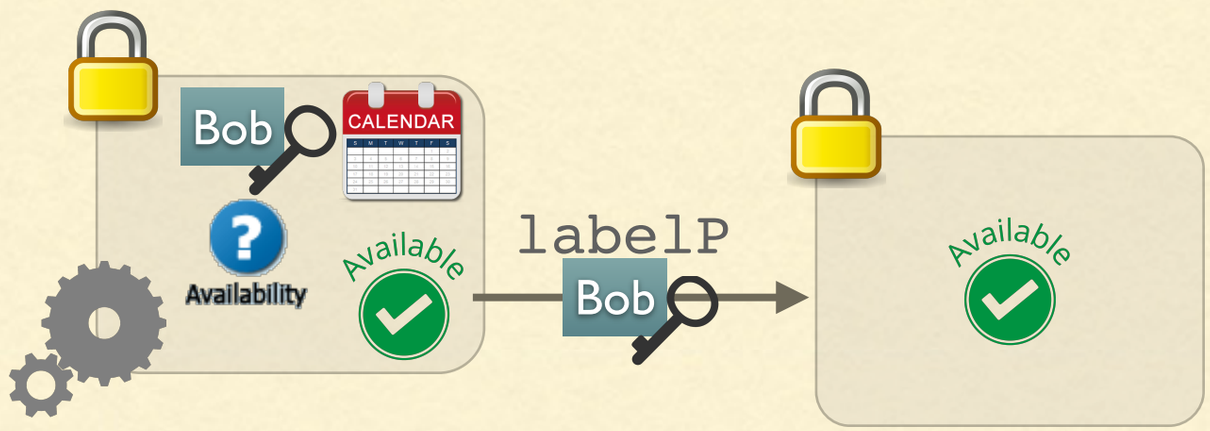
`A` was not (even partially) responsible for the current computational context performing the declassification.

Dangerous Downgrading Revisited



(Eve, Eve)

(not in Bob's study group)

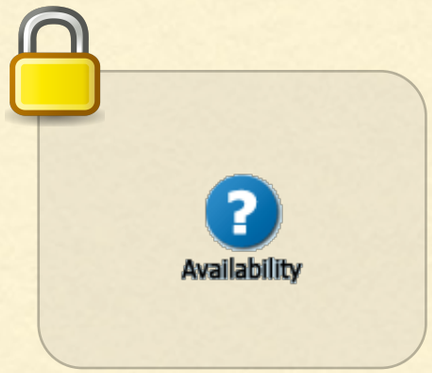


(Bob ∧ Eve, Bob ∨ Eve)

(Eve, Bob ∨ Eve)

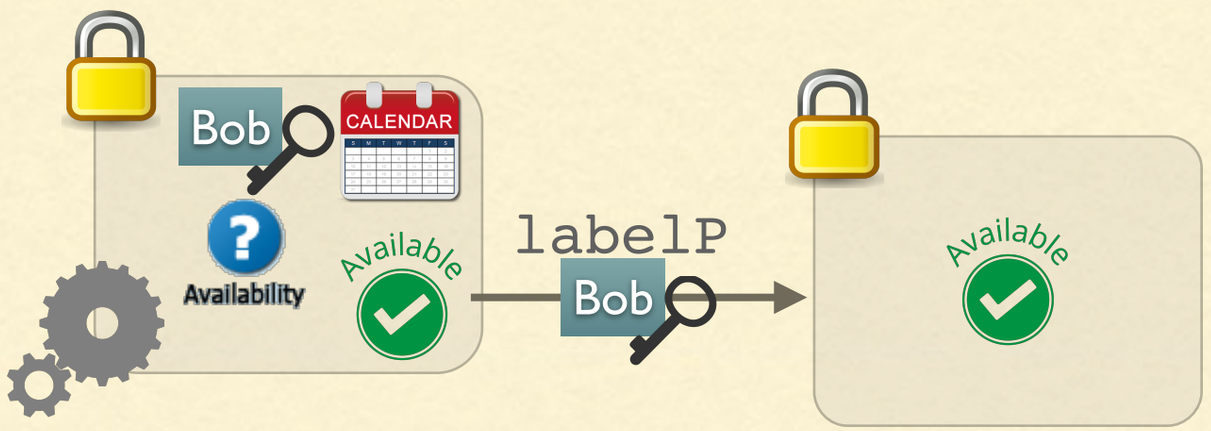
Dangerous Downgrading Revisited

L_{from}: (Bob ∧ Eve, Bob ∨ Eve)
L_{PC}: (Bob ∧ Eve, Bob ∨ Eve)
L_{to}: (Eve, Bob ∨ Eve)



(Eve, Eve)

(not in Bob's study group)



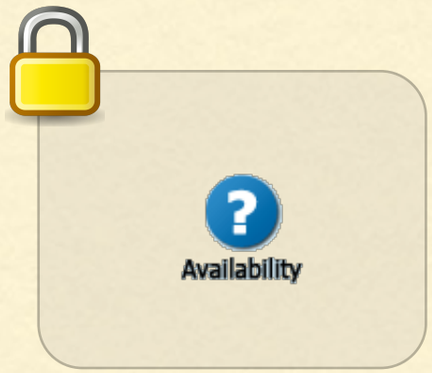
(Bob ∧ Eve, Bob ∨ Eve)

(Eve, Bob ∨ Eve)

Dangerous Downgrading Revisited

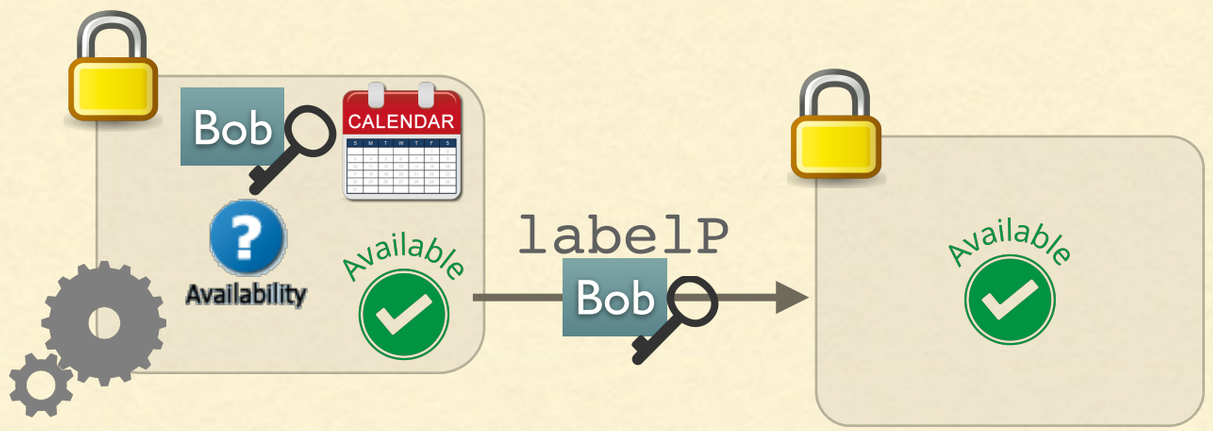
L_{from}: (Bob ∧ Eve, Bob ∨ Eve)
L_{PC}: (Bob ∧ Eve, Bob ∨ Eve)
L_{to}: (Eve, Bob ∨ Eve)

Since Eve influenced the declassification but will learn from it, declassification is not robust.



(Eve, Eve)

(not in Bob's study group)



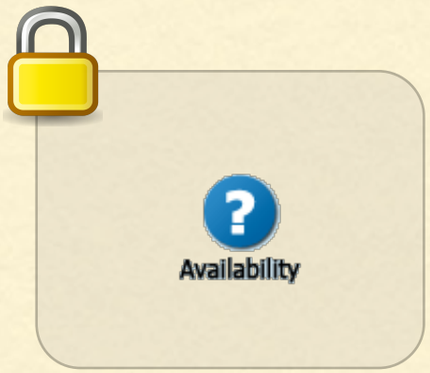
(Bob ∧ Eve, Bob ∨ Eve)

(Eve, Bob ∨ Eve)

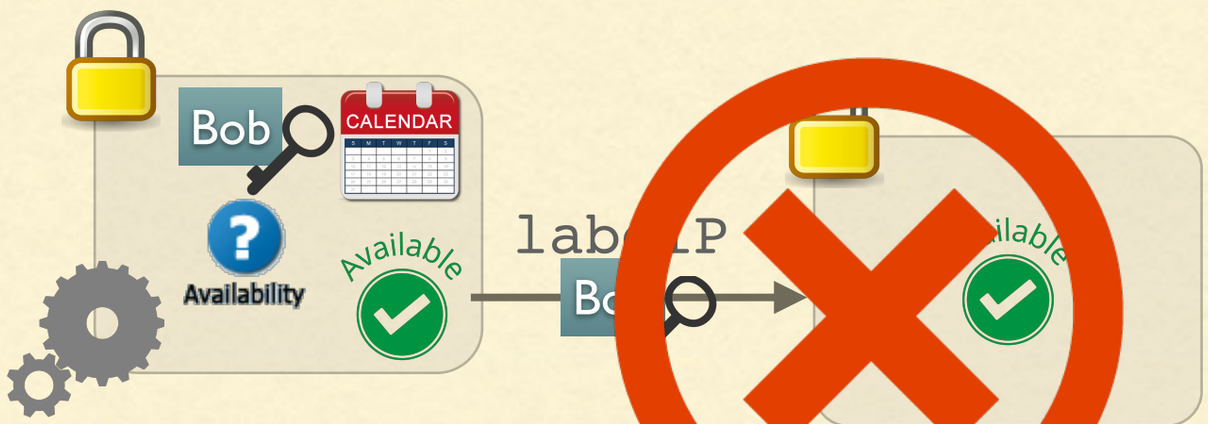
Dangerous Downgrading Revisited

L_{from}: (Bob ∧ Eve, Bob ∨ Eve)
L_{PC}: (Bob ∧ Eve, Bob ∨ Eve)
L_{to}: (Eve, Bob ∨ Eve)

Since Eve influenced the declassification but will learn from it, declassification is not robust.



(Eve, Eve)



(Bob ∧ Eve, Bob ∨ Eve)

(Eve, Bob ∨ Eve)

(not in Bob's study group)

Composing Privileges

Composing Privileges

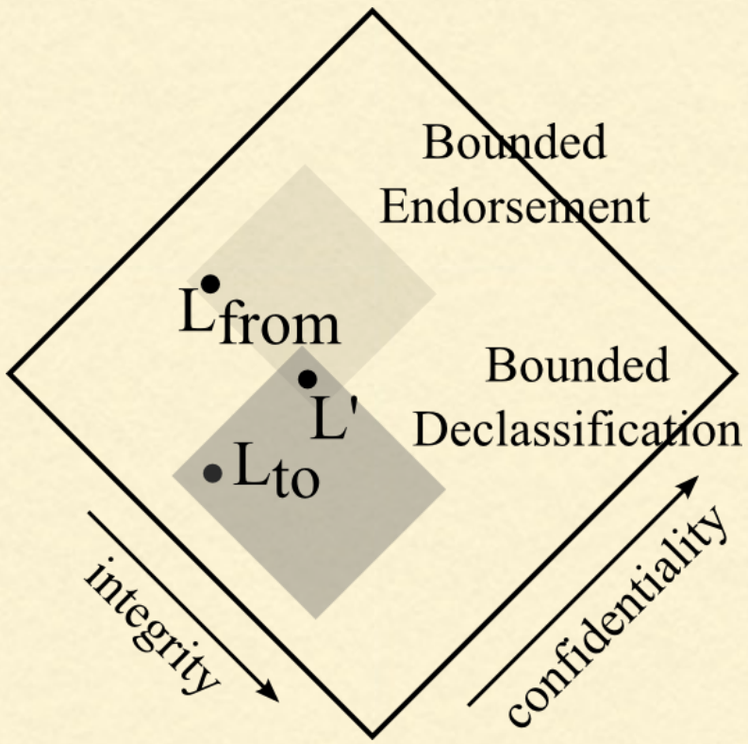
Restricted privileges can be composed with other privileges

- Bounded privilege around another bounded privilege must satisfy both bounds
- A robust declassification can be used so long as it mains some amount of secrecy (e.g., must stay within the organization)

Multiple Privileges

System may have multiple privileges available to it

- **Example**:
  - A bounded endorsement privilege can be used to go to L'
  - Then a bounded declassification privilege can go from L' to L_{to}.



Calendar Case Study

We implemented the calendar example in LIO using only restricted privileges.

Calendar Case Study

We implemented the calendar example in LIO using only restricted privileges.

- Enabled reasoning about security properties of application by looking at the restricted privileges available in the system

Calendar Case Study

We implemented the calendar example in LIO using only restricted privileges.

- Enabled reasoning about security properties of application by looking at the restricted privileges available in the system

No need to inspect how the code uses the privileges

Conclusions

- Privileges are a relatively recent and novel approach to downgrading information in IFC systems.
- Restricted privileges are a new mechanism to control downgrading in DC-labels.
- Also in the paper:
  - relationship of privileges to known semantic security guarantees
  - description of security bug in DC-labels application
  - efficient runtime checks and proofs of soundness and completeness w.r.t. formal definitions

Conclusions

- Privileges are a relatively recent and novel approach to downgrading information in IFC systems.
- Restricted privileges are a new mechanism to control downgrading in DC-labels.
- Also in the paper:
  - relationship of privileges to known semantic security guarantees
  - description of security bug in DC-labels application
  - efficient runtime checks and proofs of soundness and completeness w.r.t. formal definitions

Restricted privileges are a **declarative** mechanism for reasoning about **global** security properties without inspecting how they might be used.