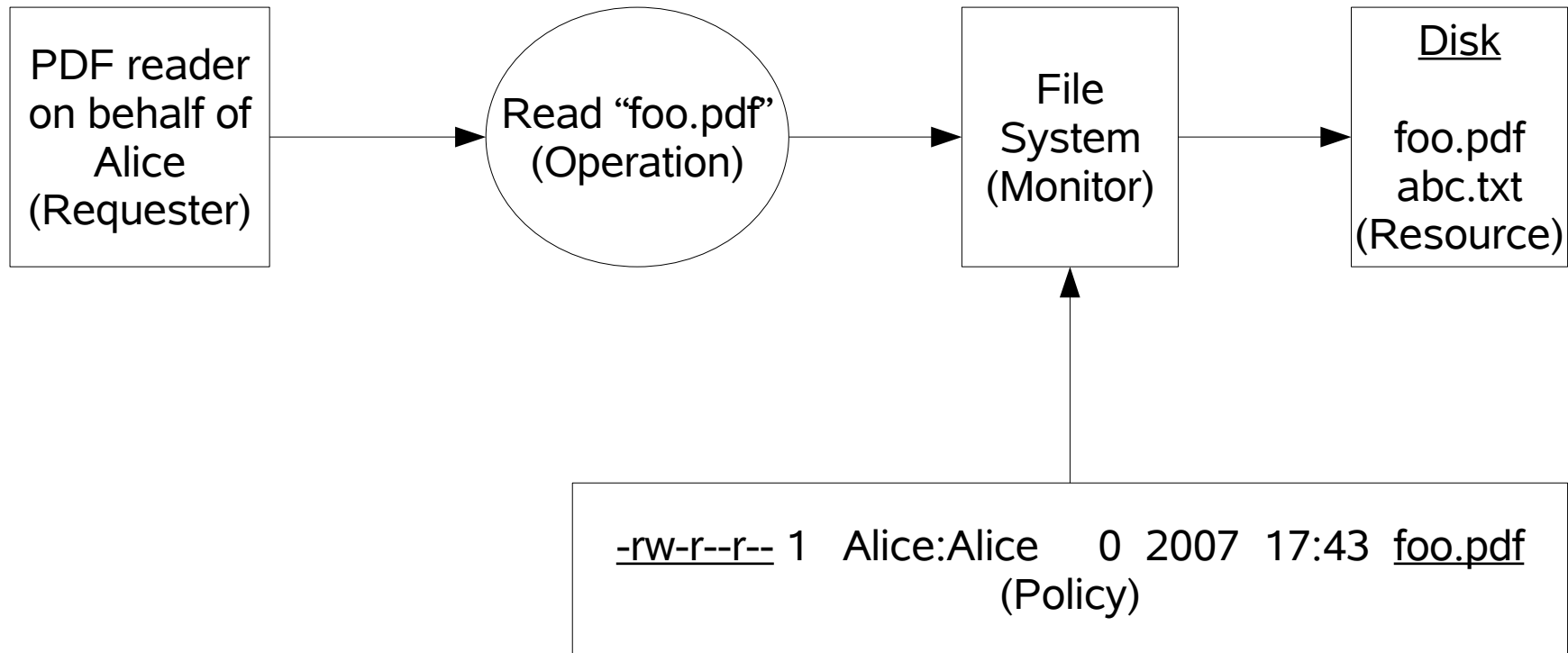


Logics for Distributed Access Control

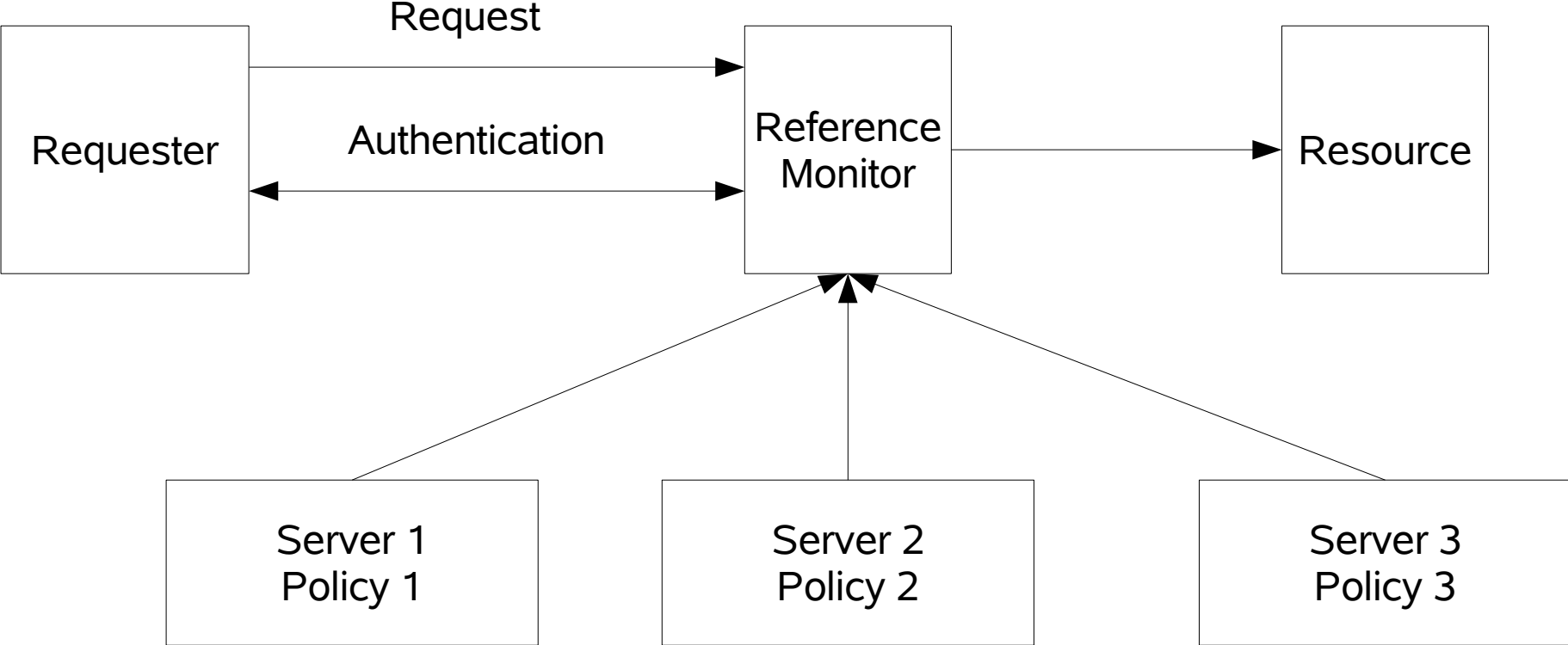
Deepak Garg

Foundations of Security and Privacy
Fall 2007

What is Access Control?

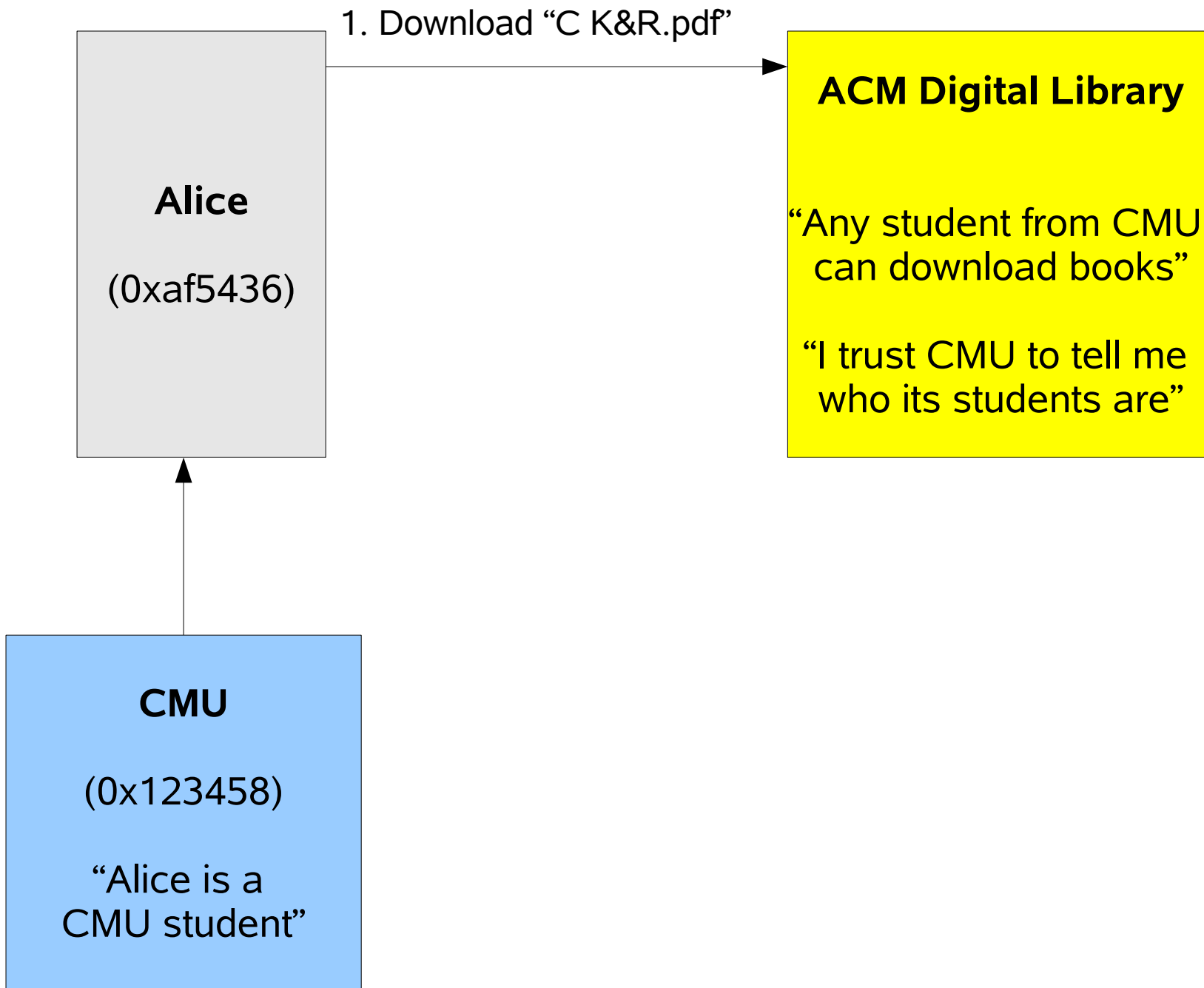


Broad Picture



Distributed Access Control Issues

- Remote requesters
 - Naming
 - CMU calls someone Alice, MIT calls someone else Alice
 - Authentication requires mechanism
 - Tokens, protocols
- Distributed policies
 - How are policies represented in a uniform manner?
 - How does the monitor combine policies of several principals?
 - How is reasoning done?



Representing Policies in a Logic

- Policy is written as formulas in the logic
 - CMU says $\text{isStudent}(\text{Alice})$
 - ACM says $\{X\}(\text{isStudent}(X) \rightarrow \text{canDownload}(X))$
 - ACM says $\{X\}((\text{CMU says isStudent}(X)) \rightarrow \text{isStudent}(X))$
- *Inference*: process of proving that a formula holds, given the policy
 - We can infer (or prove) that
(ACM says $\text{canDownload}(\text{Alice})$)
- Subject to **fixed and well defined rules**

Inference Rules (Last Week)

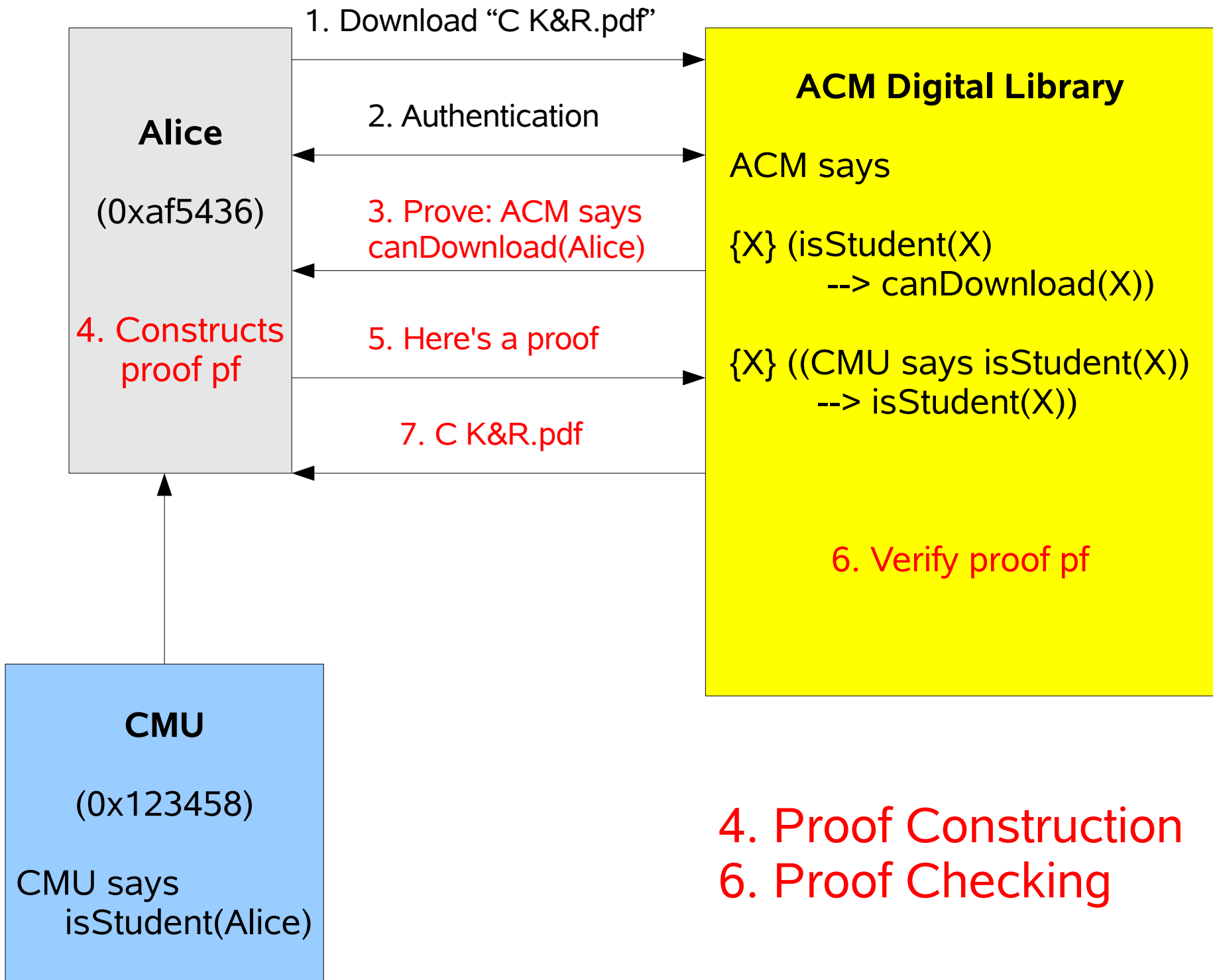
$$\frac{A \text{ in } P}{P \Rightarrow A} \text{(1)}$$

$$\frac{P, A \Rightarrow B}{P \Rightarrow A \leftrightarrow B} \text{(2)}$$

$$\frac{P \Rightarrow A \leftrightarrow B \quad P \Rightarrow A}{P \Rightarrow B} \text{(3)}$$

$$\frac{P \Rightarrow A}{P \Rightarrow \text{K says } A} \text{(4)}$$

$$\frac{P \Rightarrow \text{K says } A \quad P, A \Rightarrow \text{K says } B}{P \Rightarrow \text{K says } B} \text{(5)}$$



Rest of the Lecture ...

- What is a proof?
 - From mathematical (geometric) to logical proofs
 - Inference rules (proof rules)
- What is an access control logic?
 - Expressiveness
 - Meta-theory
- What are access control logics good for?
 - Reasoning *about* policies

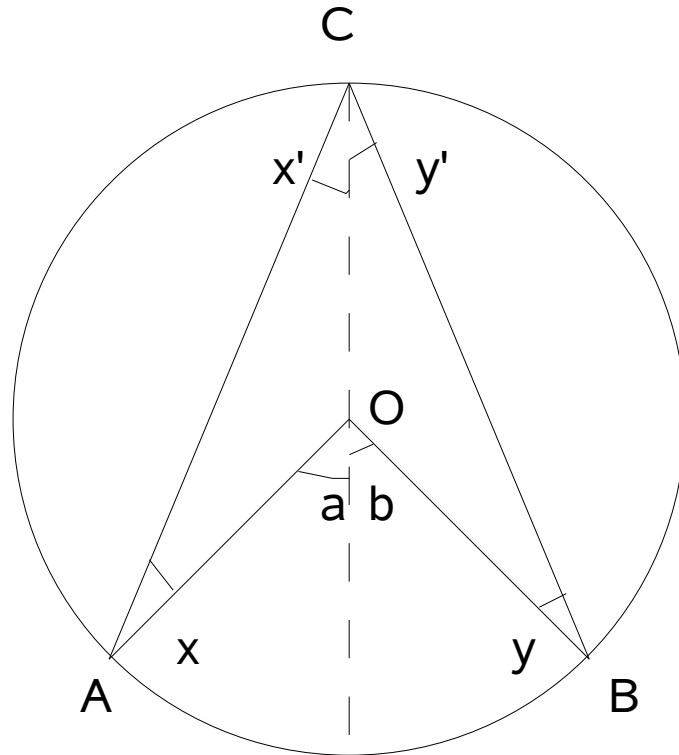
History of Access Control Logics

- Introduced by Abadi et al. [1992, 1993]
 - Objective was to model access control in OS
 - Introduced the construct (K says A)
- Used by several other proposals since
- PCA [Felten et al 1999, Bauer 2003]
- Indexed lax logic [2006]

Proof-theory: The study of logical proofs

Proofs in Geometry

Theorem A: $\text{ang}(\text{AOB}) = 2 * \text{ang}(\text{ACB})$



Facts (Axioms):

1. $\text{ang}(\text{ACB}) = x' + y'$

2. $\text{ang}(\text{AOB}) = a + b$

Lemmas:

3. $a = x + x'$ (Exterior angles)

4. $b = y + y'$ (Exterior angles)

5. $x = x'$ (Isosceles triangle)

6. $y = y'$ (Isosceles triangle)

Proof:

$$\text{ang}(\text{AOB}) = a + b \quad (2)$$

$$a + b = x + x' + y + y' \quad (3,4)$$

$$a + b = x' + x' + y' + y' \quad (5,6)$$

$$a + b = 2 * (x' + y')$$

$$\text{ang}(\text{ACB}) = x' + y' \quad (1)$$

$$\text{ang}(\text{AOB}) = 2 * \text{ang}(\text{ACB})$$

From facts 1,2 and lemmas 3-6,
we can establish theorem A

Logic vs Geometry

Theorem: ACM says canDownload(Alice)

Theorem A: $\text{ang}(\text{AOB}) = 2 * \text{ang}(\text{ACB})$

Facts (Axioms or Policy)

P =
{p1: CMU says isStudent(Alice)
p2: ACM says
 {X} (isStudent(X) --> canDownload(X))
p3: ACM says
 {X} ((CMU says isStudent(X)) -->
 isStudent(X))
}

Facts (Axioms):

1. $\text{ang}(\text{ACB}) = x' + y'$
2. $\text{ang}(\text{AOB}) = a + b$

Lemmas:

3. $a = x + x'$ (Exterior angles)
4. $b = y + y'$ (Exterior angles)
5. $x = x'$ (Isoceles triangle)
6. $y = y'$ (Isoceles triangle)

Proof-term:

M = <ACM>(let<ACM>p2' = p2 in
 let<ACM>p3' = p3 in
 aff ACM
 (p2' Alice (p3' Alice p1)))

Proof:

$$\begin{aligned} \text{ang}(\text{AOB}) &= a+b && (2) \\ a+b &= x + x' + y + y' && (3,4) \\ a+b &= x' + x' + y' + y' && (5,6) \\ a+b &= 2 * (x' + y') \\ \text{ang}(\text{ACB}) &= x' + y' && (1) \\ \text{ang}(\text{AOB}) &= 2 * \text{ang}(\text{ACB}) \end{aligned}$$

P => M : ACM says canDownload(Alice)

From facts 1,2 and lemmas 3-6,
we can establish theorem A

Logic: Formal Syntax

- Principals K
- Formulas (A, B, C, \dots)
 - P (atomic predicates)
 - $A \rightarrow B$ (implication)
 - $K \text{ says } A$ (assertion)
 - $\{x\} A$ (universal quantification)

Logic: Judgments

- $P \Rightarrow M : A$
 - Under assumptions P , we can infer A (M is the proof)
- $P \Rightarrow E : K \text{ affirms } A$
 - Under assumptions P , we can infer that K supports or believes A (E is the proof)

Inference (Proof) Rules

$$\frac{}{P, p: A \Rightarrow p: A} \text{ (use)}$$

$$\frac{P \Rightarrow M: A}{P \Rightarrow \text{aff } K \text{ } M: K \text{ affirms } A} \text{ (aff)}$$

$$\frac{P, p: A \Rightarrow M: B}{P \Rightarrow [p: A] M: A \rightarrow B} \text{ (}\rightarrow\text{ I)}$$

$$\frac{P \Rightarrow M: A \rightarrow B \quad P \Rightarrow N: A}{P \Rightarrow M N: B} \text{ (}\rightarrow\text{ E)}$$

$$\frac{P \Rightarrow E: K \text{ affirms } A}{P \Rightarrow \langle K \rangle E: K \text{ says } A} \text{ (says I)}$$

$$\frac{P \Rightarrow M: K \text{ says } A \quad P, p: A \Rightarrow E: K \text{ affirms } C}{P \Rightarrow \text{let } \langle K \rangle p = M \text{ in } E: K \text{ affirms } C} \text{ (says E)}$$

$$\frac{P \Rightarrow M: A}{P \Rightarrow \{x\} M: \{x\} A} \text{ (}\{\}\text{ I)}$$

$$\frac{P \Rightarrow M: \{x\} A}{P \Rightarrow M K: A[K/x]} \text{ (}\{\}\text{ E)}$$

ACM says canDownload(Alice)

$$\frac{}{P, p : A \Rightarrow p : A} \text{ (use)}$$

$$\frac{P \Rightarrow M : A}{P \Rightarrow \text{aff } K \text{ } M : K \text{ affirms } A} \text{ (aff)}$$

$$\frac{P \Rightarrow M : A \dashrightarrow B \quad P \Rightarrow N : A}{P \Rightarrow M N : B} \text{ (}\dashrightarrow\text{ E)}$$

Proof:
 $M = \langle \text{ACM} \rangle (\text{let} \langle \text{ACM} \rangle p2' = p2 \text{ in}$
 $\quad \text{let} \langle \text{ACM} \rangle p3' = p3 \text{ in}$
 $\quad \text{aff ACM}$
 $\quad (p2' \text{ Alice } (p3' \text{ Alice } p1)))$

$$\frac{P \Rightarrow M : \{x\} A}{P \Rightarrow M K : A[K/x]} \text{ (}\{ \} \text{ E)}$$

$$\frac{P \Rightarrow E : K \text{ affirms } A}{P \Rightarrow \langle K \rangle E : K \text{ says } A} \text{ (says I)}$$

$$\frac{P \Rightarrow M : K \text{ says } A \quad P, p : A \Rightarrow E : K \text{ affirms } C}{P \Rightarrow \text{let } \langle K \rangle p = M \text{ in } E : K \text{ affirms } C} \text{ (says E)}$$

$P = \{ p1: \text{CMU says isStudent(Alice)}, p2: \text{ACM says } \{x\} (\text{isStudent}(x) \rightarrow \text{canDownload}(x)), p3: \text{ACM says } \{x\} ((\text{CMU says isStudent}(x)) \rightarrow \text{isStudent}(x)) \}$

$P' = P, p2': \{x\} (\text{isStudent}(x) \rightarrow \text{canDownload}(x)), p3': \{x\} ((\text{CMU says isStudent}(x)) \rightarrow \text{isStudent}(x))$

$P'' = P, p2': \{x\} (\text{isStudent}(x) \rightarrow \text{canDownload}(x))$

$P \Rightarrow p2 : \text{ACM says } \{x\} (\text{isStudent}(x) \rightarrow \text{canDownload}(x))$

$P \Rightarrow p3: \text{ACM says } \{x\} ((\text{CMU says isStudent}(x)) \rightarrow \text{isStudent}(x))$

$P' \Rightarrow p2': \{x\} (\text{isStudent}(x) \rightarrow \text{canDownload}(x))$

$P' \Rightarrow p2' \text{ Alice}: (\text{isStudent}(\text{Alice}) \rightarrow \text{canDownload}(\text{Alice}))$

$P' \Rightarrow p3': \{x\} ((\text{CMU says isStudent}(x)) \rightarrow \text{isStudent}(x))$

$P' \Rightarrow p3' \text{ Alice}: ((\text{CMU says isStudent}(\text{Alice})) \rightarrow \text{isStudent}(\text{Alice}))$

$P' \Rightarrow p1: \text{CMU says isStudent}(\text{Alice})$

$P' \Rightarrow p3' \text{ Alice } p1: \text{isStudent}(\text{Alice})$

$P' \Rightarrow p2' \text{ Alice } (p3' \text{ Alice } p1): \text{canDownload}(\text{Alice})$

$P' \Rightarrow \text{aff ACM } (p2' \text{ Alice } (p3' \text{ Alice } p1)): \text{ACM affirms canDownload}(\text{Alice})$

$P'' \Rightarrow \text{let } \langle \text{ACM} \rangle p3' = p3 \text{ in}$

$\text{aff ACM } (p2' \text{ Alice } (p3' \text{ Alice } p1)): \text{ACM affirms canDownload}(\text{Alice})$

$P \Rightarrow \text{let } \langle \text{ACM} \rangle p2' = p2 \text{ in}$

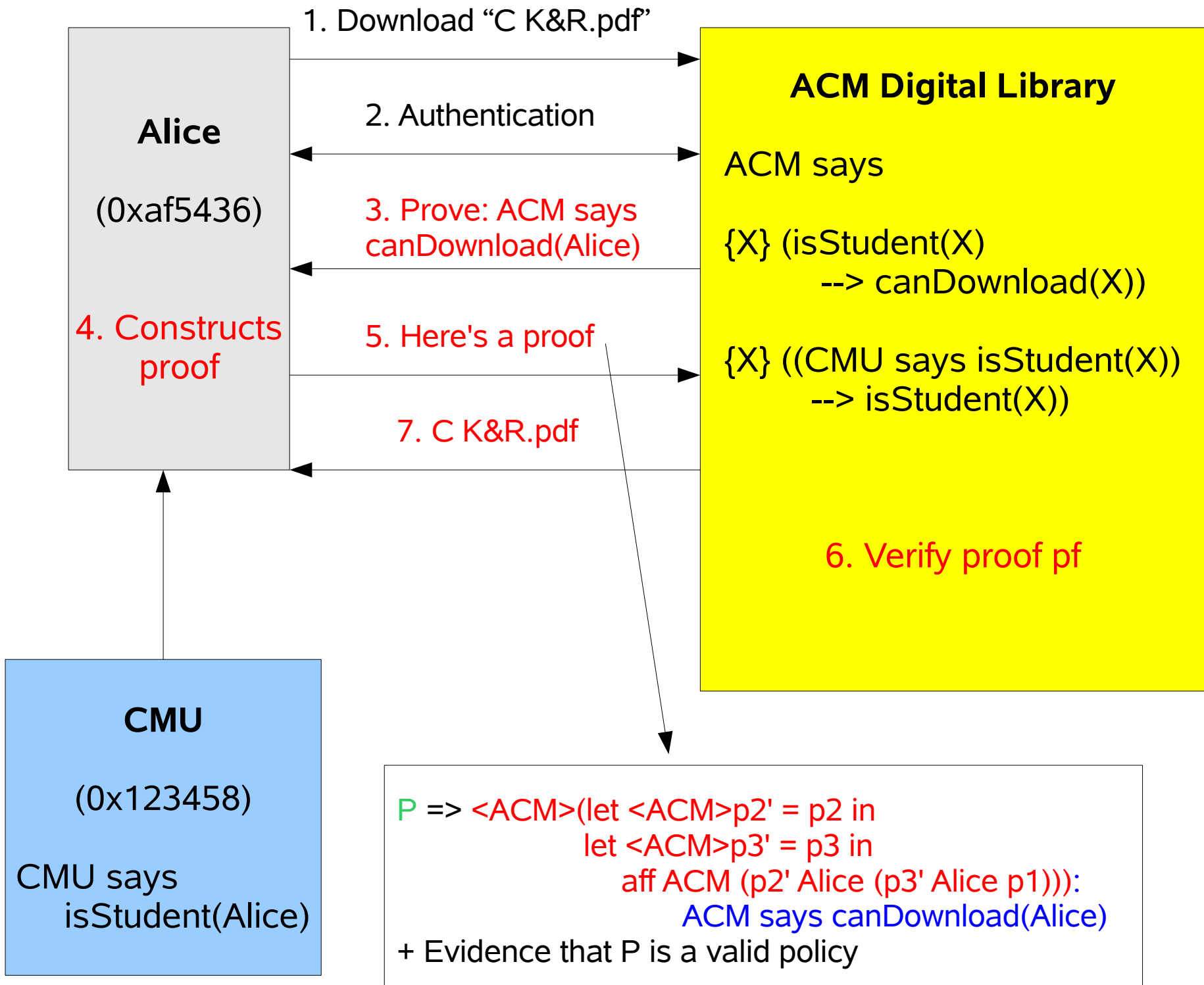
$\text{let } \langle \text{ACM} \rangle p3' = p3 \text{ in}$

$\text{aff ACM } (p2' \text{ Alice } (p3' \text{ Alice } p1)): \text{ACM affirms canDownload}(\text{Alice})$

$P \Rightarrow \langle \text{ACM} \rangle (\text{let } \langle \text{ACM} \rangle p2' = p2 \text{ in}$

$\text{let } \langle \text{ACM} \rangle p3' = p3 \text{ in}$

$\text{aff ACM } (p2' \text{ Alice } (p3' \text{ Alice } p1))): \text{ACM says canDownload}(\text{Alice})$



Verifying a Proof

- How does the server verify Alice's proof?

$P \Rightarrow \langle \text{ACM} \rangle (\text{let } \langle \text{ACM} \rangle p2' = p2 \text{ in}$
 $\quad \text{let } \langle \text{ACM} \rangle p3' = p3 \text{ in}$
 $\quad \text{aff ACM } (p2' \text{ Alice } (p3' \text{ Alice } p1))) : \text{ACM says canDownload(Alice)}$

- There is only one rule that can create $\langle \text{ACM} \rangle \dots$

$$\frac{P \Rightarrow E : K \text{ affirms } A}{P \Rightarrow \langle K \rangle E : K \text{ says } A} \text{---(says I)}$$

- Therefore, it must be the case that

$P \Rightarrow$ $(\text{let } \langle \text{ACM} \rangle p2' = p2 \text{ in}$
 $\quad \text{let } \langle \text{ACM} \rangle p3' = p3 \text{ in}$
 $\quad \text{aff ACM } (p2' \text{ Alice } (p3' \text{ Alice } p1))) : \text{ACM affirms canDownload(Alice)}$

Verifying a Proof

- This new proof is smaller!
- Process repeats until we hit names, i.e., judgments of the form:

$$P \Rightarrow p : A$$

- There is only one rule that applies (use)

$$\frac{\text{----- (use)}}{P, p : A \Rightarrow p : A}$$

- Check that $p : A$ is in the assumptions
- Time required is **linear** in the size of proof

What about P ?

- After verifying $P \Rightarrow M : A$, server knows:
 - “Assuming policy P , we can infer A because M is a correct proof”
 - Alice can download file if the policy P is actually valid
- Certificates!
- $P = \{p1: \text{CMU says isStudent(Alice)}, \dots\}$
- CMU signs (with its signing key)
“isStudent(Alice)”
- Alice submits certificates with judgment

Meta-theory of Access Control Logic

Unused Assumptions

- Do I have to use all the policy rules?
 - I established $P \Rightarrow M : B$
 - I accidentally made some assumptions in P which I never used!
 - Is my proof valid? Do I need to certify unused assumptions?
- Yes and No! Meta-theorems called strengthening and weakening
 - If $P, x : A \Rightarrow M : B$ and x not in M , then $P \Rightarrow M : B$
 - If $P \Rightarrow M : B$ then $P, x : A \Rightarrow M : B$

Lemmas

- What about lemmas?
 - Suppose I can establish $P, x: A \Rightarrow M : B$
 - Assuming lemma A, we can prove theorem B
 - I can separately establish A. $P \Rightarrow N : A$
 - How do I combine these?
- Meta-theorem called substitution
 - If $P \Rightarrow N : A$ and $P, x: A \Rightarrow M : B$ then
$$P \Rightarrow M[N/x] : B$$
 - $M[N/x]$ obtained by putting proof N for assumption x everywhere in M

Important Theorems

- $P \Rightarrow ? : K \text{ says } K \text{ says } A$ iff $P \Rightarrow ? : K \text{ says } A$
 - Repeating (K says) has no effect
- $K \text{ says } A$ does not imply A (in general)
- A implies $K \text{ says } A$
- There is no proof of false (consistency)
- $K \text{ says false}$ does not imply false
 - Principals cannot make the logic inconsistent

Proof normalization

- Every proof **M** can be reduced to a “simplest” canonical form
- Basic idea: remove circuitous routes (e.g. introduction of \rightarrow immediately followed by elimination)
- Reduced proof is easier to read, sometimes gives insights
- Possible to directly construct canonical proofs
 - Method called “sequent calculus”
 - Also simplifies proof search

Expressing Common Access Control Idioms

User groups

- We have a group called “priv”
- Every member of the group can access all files
- How do we represent this?

- Make a predicate `member(K, G)`
 - K is a member of group G
- Add a list of certificates to include members:
 - admin says `member(Alice, priv)`
- Tie authorization to the group:
 - admin says `{x} (member(x, priv) --> mayaccess(x))`

Delegation

- Allowing another principal to state some facts on your behalf
- Encode as *Alice says ((Bob says A) --> A)*
 - Whenever *(Bob says A)*, *(Alice says A)*
- Example:
 - ACM says {x} ((CMU says isStudent(X)) --> isStudent(X))

Full Delegation (Abadi et al.)

- Allowing another principal to state **any** facts on your behalf
- Cannot be encoded in the logic so far
- Add new connective (**Alice speaks for Bob**)
- Property:
 - Whenever (Alice speaks for Bob) and (Alice says A), then (Bob says A)
- Dangerous construct: does Bob know what authority he delegated?
- Useful for modelling keys: **k speaks for (owner(k))**

Analyzing Policies

Simple vs Complicated

- Policy:
 - CMU says $\text{isStudent}(\text{Alice})$
 - ACM says $\{X\}(\text{isStudent}(X) \rightarrow \text{canDownload}(X))$
 - ACM says $\{X\}((\text{CMU says isStudent}(X)) \rightarrow \text{isStudent}(X))$
- Simple questions:
 - Can Alice download a file? Yes
 - Can Bob download a file? No
 - Such questions can be answered with a theorem prover

Simple vs Complicated

- Policy:
 - CMU says $\text{isStudent}(\text{Alice})$
 - ACM says $\{X\}(\text{isStudent}(X) \rightarrow \text{canDownload}(X))$
 - ACM says $\{X\}((\text{CMU says isStudent}(X)) \rightarrow \text{isStudent}(X))$
- Complicated question
 - Suppose **Alice says isStudent(K)** for some K
 - Will this allow K to download a file?
 - Cannot be answered with a theorem prover
 - Need some better analysis
 - Proof theory!

Non-interference Theorems

- Theorem of the following form:

If P , $x: A \Rightarrow M : B$, and

<some verifiable condition on P , A , B >,

then there is some M' such that $P \Rightarrow M' : B$

- Establishes that assumption A is useless (it does not interfere with theorem B)
- Can be established for generic formulas A , B
- Make heavy use of proof-theory

Non-interference Theorems

- Basic non-interference:

If P , $x: K$ says $A \Rightarrow M : B$, and

$\langle K$ does not occur in P , B and P , B do not have $\{x\}$,
then there is some M' such that $P \Rightarrow M' : B$

- Unrelated principals cannot affect consequences of policies

Non-interference for ACM Example

- If P is the policy in the ACM example, and
 $P, x: \text{Alice says isStudent}(K) \Rightarrow$
 $M : \text{ACM says canDownload}(K'),$
then for some $M', P \Rightarrow M' : \text{ACM says canDownload}(K')$
- “Alice has no jurisdiction over predicate isStudent”

Other Issues

First-order vs Higher-order

- Logic in Grey (previous lecture) is higher-order
 - Meta-theorems are hard to establish
 - Few are known really
 - Little hope of proving non-interference
- This lecture's logic is first-order
 - Meta-theory simpler
 - Sufficiently expressive for applications in Grey

Intuitionistic vs Classical Logic

- Logic here is intuitionistic (constructive)
 - Does not allow proof by contradiction
- Several other proposals are classical
- Classical logic does not really work with our “says”
 - $(K \text{ says } A)$ implies $(K \text{ says false})$ or A
- Deeper issue related to how much information proofs give

Modelling Time

- Most signed certificates in practice have validity
- How does the logic reflect this?
 - How do I make sure my proof is valid?
- Grey approach:
 - $P \Rightarrow M : A$ is valid during intervals where each certificate validating P is valid
- Alternate approach: explicitly introduce time in the logic
 - More expressive

Conclusion

- Access control logics are a rich formalism
- Encode policies and reason with them
- Enforce policies (PCA)
- Analyze policies (Non-interference)
- Proofs are hard to construct (in general), easy to verify
- Constructive, first-order logic suffices
 - Little justification for classical, higher-order logic