CS 5432:
Authentication Logics

Fred B. Schneider
Samuel B Eckert Professor of Computer Science

Department of Computer Science
Cornell University
Ithaca, New York 14853
U.S.A.

Goals

Facility in reasoning with **says** and **speaksfor**

- Knowledge of CAL axioms and inference rules.
- Formalization of protocol goals in CAL.
- Formalization of protocol description in CAL.

N.b. Comfort in formal logics also will be useful for defining type systems for information flow.

Overview

- Why formalize? Applicability of Authentication Logics.
- Logic refresher (with apologies)
  - Formulas, Theorems, Interpretations, …
- CAL
  - Formulas
  - Interpretations
  - Compound Principals
- Accountability
- Credentials and certificates
- Applications

Overview

- Why formalize? Applicability of Authentication Logics.
- Logic refresher (with apologies)
  - Formulas, Theorems, Interpretations, …
- CAL
  - Formulas
  - Interpretations
  - Compound Principals
- Accountability
- Credentials and certificates
- Applications

What is a Formal Logic?

- A language of **formulas**.
  - Mechanically checkable whether a string is a formula.
- A subset of formulas called **axioms**.
- A set of **inference rules**, where **conclusion** C is mechanical transformation of **hypotheses** $P_1, P_2, \ \ldots, \ \ P_n$

$$\frac{P_1, P_2, \ \ \ldots, \ \ \ P_n}{C}$$

A **proof** is a sequence of formulas, each is an axiom or the conclusion of an inference rule whose premises appeared earlier. A **theorem** is <u>any</u> line in a proof.

Logic Example: Pqa [Hofstadter]

Formulas: $\alpha \, \mathrm{P} \, \beta \, \mathrm{Q} \, \gamma$ where α, β, γ denote aa…

Axioms

- Axiom 1: a P a Q aa
- Axiom 2: aa P a Q aaa

Inference rule

$$\frac{\alpha \, \mathrm{P} \, \beta \, \mathrm{Q} \, \gamma \, , \quad \delta \, \mathrm{P} \, \psi \, \mathrm{Q} \, \phi}{\alpha \, \delta \, \mathrm{P} \, \beta \, \psi \, \mathrm{Q} \, \gamma \, \phi}$$

PQa Proof Example

1. a P a Q aa *Axiom 1*
2. aa P a Q aaa *Axiom 2*
3. aaa P aa Q aaaaa *Inference rule: 1,2*
4. aaaa P aaa Q aaaaaaa *Inference rule: 1,3*

Assigning Meaning to Formulas

$I \vDash F$

- $\vDash$ (read: models) is a relation between statements I (aka "structures") and formulas F of the logic.
- If $I \vDash F$ holds then I is called a **model** for formula F .

- F is **valid** (written $\vDash F$): $I \vDash F$ holds in all I.
- F is **satisfiable**: $I \vDash F$ holds for some I.

Mechanics with Semantics

Theorems are mechanically derived. Yet they can reveal truths about reality…

- Logic is **sound**: $I \models F$ holds and F is a theorem implies I is a true statement.
  - $\text{Thms} \subseteq \text{Facts}$

- Logic is **complete**: I is a true statement and $I \models F$ holds implies F is a theorem.
  - $\text{Facts} \subseteq \text{Thms}$

Meaning(s) for PQa

Interpretation 1:

- $|\alpha| + |\beta| = |\gamma| \;\vDash\; \alpha \,\mathrm{P}\, \beta \,\mathrm{Q}\, \gamma$

Sound?

Complete?

Meaning(s) for PQa

Interpretation 1:

- $|\alpha| + |\beta| = |\gamma| \vDash \alpha \text{ P } \beta \text{ Q } \gamma$

Interpretation 2:

- $|\alpha| + |\beta| \geq |\gamma| \vDash \alpha \text{ P } \beta \text{ Q } \gamma$

Sound?

Complete?

Proof Styles

Hilbert Style:

1. a P a Q aa *Axiom 1*
2. aa P a Q aaa *Axiom 2*
3. aaa P aa Q aaaaa *Inference rule: 1,2*
4. aaaa P aaa Q aaaaaaa *Inference rule: 1,3*

Proof Styles

Derivation Tree: *Leaves must be axioms.*

$$\dfrac{\dfrac{\mathrm{a\,PaQaa,\;\; aaPaQaaa}}{\mathrm{aaaPaaQaaaaa}}, \qquad \dfrac{\mathrm{a\,PaQaa,} \quad \dfrac{\mathrm{a\,PaQaa,\;\; aaPaQaaa}}{\mathrm{aaaPaaQaaaaa}}}{\mathrm{aaaaPaaaQaaaaaaa}}}{\mathrm{aaaaaaaPaaaaaQaaaaaaaaaaaa}}$$

Hilbert Style:

1. a P a Q aa *Axiom 1*
2. aa P a Q aaa *Axiom 2*
3. aaa P aa Q aaaaa *Inference rule: 1,2*
4. aaaa P aaa Q aaaaaaa *Inference rule: 1,3*

Proof Styles

Equational Style (not always possible)

$\neg P \wedge (P \Rightarrow Q)$

= ⟨ defn of ⇒: Implication Laws (2.22a)⟩

$\neg P \wedge (\neg P \vee Q)$

= ⟨ distribution of ∧ over ∨: Distributive Laws (2.16b) ⟩

$(\neg P \wedge \neg P) \vee (\neg P \wedge Q)$

= ⟨ identity of ∧: And-Simplification Law (2.26a) ⟩

$(\neg P) \vee (\neg P \wedge Q)$

= ⟨ absorption. Or-Simplification (2.25d) ⟩

$\neg P$

Proof Styles (not)

Proof: "*We know 1+1=2. We also know that 2+1 = 3. Adding equals to equals produces (2+1)+(1+1)=(3+2). That can be formalized* as

aaa P aa Q aaaa

…"

- Explanation of how to get formal proof? (Not)
- This proof is reasoning about models but using the language of the logic.

Sequents

$F_1, F_2, \ \dots\ , \ F_n \vdash_L F$ is called a **sequent**.

Asserts that F could be proved using logic L if formulas $F_1, F_2, \ \dots\ , \ F_n$ were made axioms.

- Derivation tree with $F_1, F_2, \ \dots\ , \ F_n$ as leaves.
- In <u>some</u> logics, sequents are formulas and there is an inference rule:

$$\frac{F_1, F_2, \ \dots\ , \ F_n \vdash F}{\vdash F_1 \wedge F_2 \wedge \ \dots \wedge F_n \Rightarrow F}$$

Model Checking

Given a formula F, identify a set of "critical" models $I_1, I_2, \ldots, I_n$.

- Check $I_i \models F$ (only) for critical models $I_1, I_2 \ldots, I_n$.
    - Potentially intractable computation.
    - Often requires restriction to finite state space.
- Conclude $\models F$

Example: Using a "truth table" in propositional logic.

Overview

- Why formalize? Applicability of Authentication Logics.
- Logic refresher (with apologies)
  - Formulas, Theorems, Interpretations, …
- CAL
  - Formulas
  - Interpretations
  - Compound Principals
- Accountability
- Credentials and certificates
- Applications

CAL

Language:

C ::= F (F *a formula of First-order Predicate Logic*)
| P **says** C
| P′ **speaksfor** P
| P′ **speaks** x:C **for** P
| C ∧ C′
| C ∨ C′
| C ⇒ C′

N.b. ¬C: (C ⇒ false)

Models for CAL

$\langle \sigma , \omega \rangle \vDash \mathrm{C}$:

- σ is a state. It maps variables to values.
  - $\langle \sigma , \omega \rangle \vDash \mathrm{F}$ iff $\sigma \vDash_{Pred} \mathrm{F}$ (for pred logic F)
- ω(P) is the set of beliefs principal P has.
  - $\langle \sigma , \omega \rangle \vDash$ P **says** C iff $\mathrm{C} \in \omega(\mathrm{P})$
  - $\langle \sigma , \omega \rangle \vDash$ P' **speaksfor** P iff $\omega(\mathrm{P'}) \subseteq \omega(\mathrm{P})$

ω(P) called the **worldview** of P

Contents of $\omega(\cdot)$?

Requirement: A trustworthy P issues a credential conveying P **says** C only if C $\in \omega($ P $)$.

Conservative Approximation for $\omega($ P $)$.

- $\omega($ P $)$ contains some initial beliefs Init_P
- $\omega($ P $)$ is closed under logical consequence.
  - Logical consequence conservatively models everything that any program could deduce from local state and beliefs.

Inconsistent Beliefs

P might hold beliefs: B and $\neg$B (aka B $\Rightarrow$ false)

- P received inconsistent credentials.
- P read the state at two different times.
- P executed a buggy or malicious program.

P then cannot be trusted -- it holds all beliefs:

1. B
2. B $\Rightarrow$ false
3. False
4. B′

CAL Inference Rules: **says**

$$\frac{\vdash_{CAL}\ C}{P\ \textbf{says}\ C} \qquad \frac{P\ \textbf{says}\ C}{P\ \textbf{says}\ (\,P\ \textbf{says}\ C)} \qquad \frac{P\ \textbf{says}\ (\,P\ \textbf{says}\ C)}{P\ \text{says}\ C}$$

$$\frac{P\ \textbf{says}\ (C \Rightarrow C')}{(P\ \textbf{says}\ C\,) \Rightarrow (\,P\ \textbf{says}\ C')}$$

Example CAL Proof (1)

$$P \ \textbf{says} \ C, \qquad P \ \textbf{says} \ (\ C \Rightarrow C'\ )$$

Example CAL Proof (2)

$$P \textbf{ says } C, \quad \frac{P \textbf{ says } (\, C \Rightarrow C' \,)}{(P \textbf{ says } C) \Rightarrow (P \textbf{ says } C')}$$

Example CAL Proof (3)

$$\frac{P\ \textbf{says}\ C, \quad \dfrac{P\ \textbf{says}\ (\ C \Rightarrow C'\ )}{(P\ \textbf{says}\ C) \Rightarrow (P\ \textbf{says}\ C')}}{P\ \textbf{says}\ C'}$$

CAL Inference Rules: speaksfor

$$\frac{P \textbf{ says } (P' \textbf{speaksfor } P)}{P' \textbf{speaksfor } P} \text{ hand-off}$$

$$\frac{P' \textbf{speaksfor } P}{(P' \textbf{says } C) \Rightarrow (P \textbf{ says } C)}$$

$$\frac{P \textbf{ speaksfor } P', P' \textbf{speaksfor } P''}{P \; \boldsymbol{speaksfor} \; P''}$$

Inherited Inconsistency in CAL?

Can worldviews for different principals cause some principal to have inconsistent beliefs?

- P **says** C and P **says** ¬C -vs-
- P **says** C and P′ **says** ¬C , where
  - P′ **speaksfor** P?
  - No delegation to P′ by P?

CAL Non-Interference

Set of principals is <u>independent</u> if no element makes a delegation to another element.

Thm: For $P \in IP$, a set of independent principals:

$C_1, \ldots, C_m \vdash_{CAL} P$ **says** false

iff

$D_1, \ldots, D_n \vdash_{CAL} P$ **says** false

where no D_i includes "P_j **says** …" for $P_j \in IP$-{P}.

Unrestricted Delegation

$$\frac{P' \textbf{ says } C, \quad \dfrac{P' \textbf{ speaksfor } P}{(P' \textbf{ says } C) \Rightarrow (P \textbf{ says } C)}}{P \textbf{ says } C}$$

- **Warning**: P inherits beliefs from any principal that was delegated to.
- P trusting P' means
  - P adopts all beliefs of P'
  - P also adopts beliefs of any principal P' trusts (transitive).

Why Delegate?

Transitivity of delegation allows clients to be ignorant of the implementation details of services the clients invoke.

- Transitive delegations are made by implementation of service to lower-level services.
- Transitive delegations are hidden from clients.

Restricted Delegation

$$\frac{P' \textbf{ speaks } x\colon C \textbf{ for } P}{(P' \textbf{ says } C[x \coloneqq \tau]) \Rightarrow (P \textbf{ says } C[x \coloneqq \tau])}$$

Example:

CS **says** $Major(Alice)$

CS **says** $\neg Major(Alice)$

CU **says** (CS **speaksfor** CU) 😢

CU **says** (CS **speaks** $x\colon Major(x)$ **for** CU) 🙂

… CU does not inherit $\neg Major(x)$ from CS

Compound Principals

- Every principal P has a worldview $\omega(P)$.
- Compound principals combine worldviews from multiple principals to obtain a worldview for the compound principal.
- Example:
  - $P \wedge Q$: $\omega(\ P \wedge Q\ )$: $\omega(P) \cap \omega(Q)$

Useful Compound Principals

- Subprincipals of P: P.x
- Groups G = { P_1, P_2, … P_n }

Subprincipals

For any term η:

$$\frac{}{P\ \mathbf{speaksfor}\ P.\eta}$$

$$\frac{\eta = \eta'}{P.\eta\ \mathbf{speaksfor}\ P.\eta'}$$

Use of Subprincipals

- Any belief of P is attributed to $P.x$ for any x.
  - **Hack**: Employ $P.\epsilon$ for beliefs by P that should not be attributed to other sub-principals of P.
- If L *implements* H then H is a subprincipal of L.
  - **Example**: HW implements OS, so HW.OS is the principal that corresponds to the operating system.

Implements: CAL Analysis

L implements H, so H is a subprincipal of L.

- L **says** $(H$ **says** $C)$
- L **speaksfor** H

$$L \textbf{ says } (H \textbf{ says } C),\ \frac{L \textbf{ speaksfor } H}{\big(L \textbf{ says } (H \textbf{ says } C)\big) \Rightarrow (H \textbf{ says } (H \textbf{ says } C)}$$

Implements: CAL Analysis

L implements H, so H is a subprincipal of L.

- L **says** $(H$ **says** $C)$
- L **speaksfor** H

$$\frac{L \textbf{ says } (H \textbf{ says } C), \quad \dfrac{L \textbf{ speaksfor } H}{\big(L \textbf{ says } (H \textbf{ says } C)\big) \Rightarrow (H \textbf{ says } (H \textbf{ says } C)}}{\dfrac{H \textbf{ says } (\, H \textbf{ says } C)}{H \textbf{ says } C}}$$

Group Principals

A **group** is defined by a finite enumeration of its member principals. $G = \{ P_1 , P_2 , \ldots P_N \}$

- Conjunctive Groups

$$\frac{P_i \textbf{ says } C, \textbf{for every } P_i \in G}{P_G \textbf{ says } C}$$

$$\frac{P_G \; says \; C}{P \; says \; C} \qquad\qquad \frac{}{P_G \textbf{ speaksfor } P} \text{ for } P \in G$$

Group Principals

- Disjunctive Groups. Hold beliefs that any member principal holds plus deductive closure!

$$\frac{P \; says \; C}{P_G \; says \; C} \qquad \frac{}{P \; \mathbf{speaksfor} \; P_G} \text{ for } P \in G$$

$$\frac{P_G \; \mathbf{says} \; C \,, \quad P_G \; \mathbf{says} \; (C \Rightarrow C')}{P_G \; \mathbf{says} \; C'}$$

Overview

- Why formalize? Applicability of Authentication Logics.
- Logic refresher (with apologies)
  - Formulas, Theorems, Interpretations, …
- CAL
  - Formulas
  - Interpretations
  - Compound Principals
- Accountability
- Credentials and certificates
- Applications

Constructive Logics (1)

Constructive logics omit certain inference rules. In return, proofs have certain useful properties for our application domain.

- Evidence that justifies a decision is visible in the proof.
- Inferences made when there is partial information cannot become invalidated and new information becomes known.

Constructive Logics (2)

Omit all variants of the following rule:

$$\frac{}{F \lor \neg F}\text{-excluded middle}$$

So the following is not a proof:

$$\dfrac{\dfrac{F}{F \Rightarrow G} \quad \dfrac{\neg F}{\neg F \Rightarrow G} \quad \dfrac{}{F \lor \neg F}}{G}$$

... G because F holds or because $\neg F$ holds?

Constructive Logics (3)

Monotonicity wrt partial structures…

- Define $\langle \sigma , \omega \rangle \ll \langle \sigma' , \omega' \rangle$
  - σ assigns values to only some variables that σ' does
  - ω has a subset of the beliefs that ω' does, for all prins.
- **Thm**: For all CAL formulas F:

$$\langle \sigma , \omega \rangle \ll \langle \sigma' , \omega' \rangle \Rightarrow \quad (\langle \sigma , \omega \rangle \vDash F \quad \Rightarrow \quad \langle \sigma' , \omega' \rangle \vDash F\ )$$

  - F may hold before you know whether $\neg F$ does
  - F may hold even though all certificates have not been received.
  - N.b. $\neg$ (P **says** S) is not a CAL formula

Overview

- Why formalize? Applicability of Authentication Logics.
- Logic refresher (with apologies)
  - Formulas, Theorems, Interpretations, …
- CAL
  - Formulas
  - Interpretations
  - Compound Principals
- Accountability
- Credentials and certificates
- Applications

Credentials Can Convey Beliefs

k_S-**sign**(C): K_S **says** C

- Public keys are principals.
- K_S **speaksfor** S if principal S is the only agent with access to private key k_S.

A principal S can be a hash of the running code and data that was read.

Overview

- Why formalize? Applicability of Authentication Logics.
- Logic refresher (with apologies)
  - Formulas, Theorems, Interpretations, …
- CAL
  - Formulas
  - Interpretations
  - Compound Principals
- Accountability
- Credentials and certificates
- Applications

Application 1:
Public Key Infrastructure (PKI)

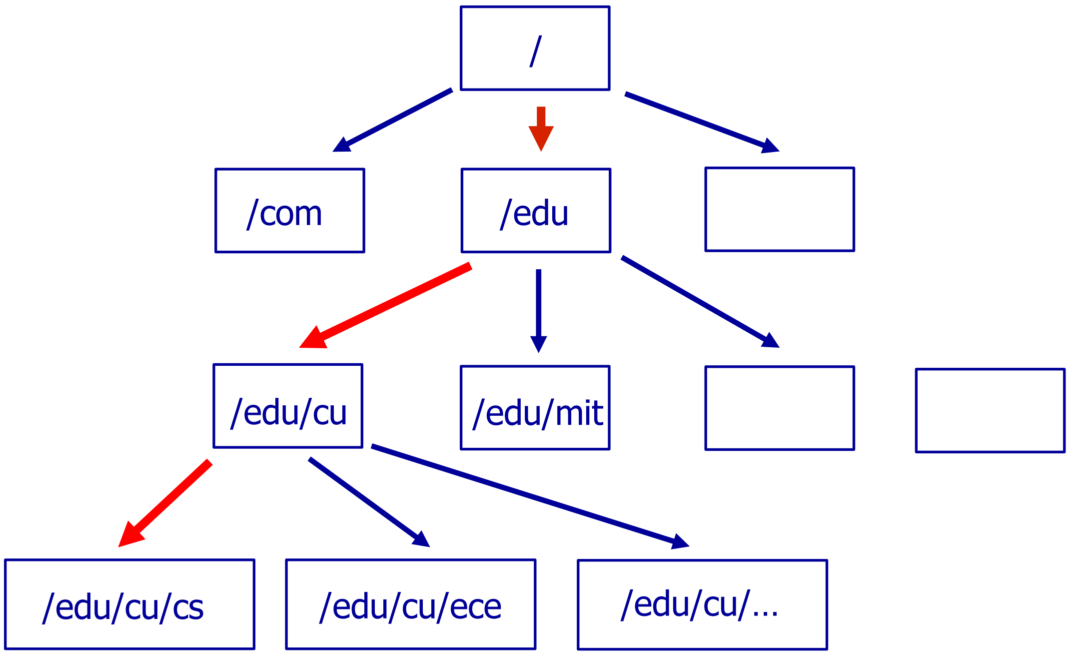
k_S-**sign**(C):

- Certificate: K_S-⟨C⟩
- CAL formalization: K_S **says** C

CAL formalization of delegation certificate:

- Certificate: K_I-⟨ϵ/com : K_{com}⟩
- CAL formalization: K_I **says** (K_{com} **speaksfor** ϵ/com)

Public Key Infrastructure (PKI)



PKI Excerpt

...
K_I-$\langle \epsilon$/com : $K_{com}\rangle$
K_I-$\langle \epsilon$/edu : $K_{edu}\rangle$
...

/

...
K_{edu}-$\langle \epsilon$/edu/cu : $K_{cu}\rangle$
K_{edu}-$\langle \epsilon$/edu/mit : $K_{mit}\rangle$
...

/edu

...
K_{cu}-$\langle \epsilon$/edu/cu/cs : $K_{cs}\rangle$
K_{cu}-$\langle \epsilon$/edu/cu/ece : $K_{ece}\rangle$
...

/edu/cu

...
K_{cs}-$\langle \epsilon$/edu/cu/cs/fbs : $K_{fbs}\rangle$
K_{cs}-$\langle \epsilon$/edu/cu/cs/la : $K_{la}\rangle$
...

/edu/cu/cs

CAL Model for PKI Excerpt

…
K_I-⟨ϵ/com : K_{com}⟩ ➡ K_I **says** (K_{com} **speaksfor** ϵ/com)
K_I-⟨ϵ/edu : K_{edu}⟩ ➡ K_I **says** (K_{edu} **speaksfor** ϵ/edu)
…

…
K_{edu}-⟨ϵ/edu/cu : K_{cu}⟩ ➡ K_{edu} **says** (K_{cu} **speaksfor** ϵ/edu/cu)
K_{edu}-⟨ϵ/edu/mit : K_{mit}⟩ ➡ K_{edu} **says** (K_{mit} **speaksfor** ϵ/edu/mit)
…

…
K_{cu}-⟨ϵ/edu/cu/cs : K_{cs}⟩ ➡ K_{cu} **says** (K_{cs} **speaksfor** ϵ/edu/cu/cs)
K_{cu}-⟨ϵ/edu/cu/ece : K_{ece}⟩ ➡ K_{cu} **says** (K_{ece} **speaksfor** ϵ/edu/cu/ece)
…

…
K_{cs}-⟨ϵ/edu/cu/cs/fbs : K_{fbs}⟩ ➡ K_{cs} **says** (K_{fbs} **speaksfor** ϵ/edu/cu/cs/fbs)
K_{cs}-⟨ϵ/edu/cu/cs/la : K_{la}⟩ ➡ K_{cs} **says** (K_{la} **speaksfor** ϵ/edu/cu/cs/la)
…

Sample Derivation

K_{fbs} **speaksfor** ϵ/edu/cu/cs/fbs

CAL Model for PKI Except

…
K_I-⟨ϵ/com : K_{com}⟩
K_I-⟨ϵ/edu : K_{edu}⟩ ➡ K_I **says** (K_{edu} **speaksfor** ϵ/edu)
…

…
K_{edu}-⟨ϵ/edu/cu : K_{cu}⟩ ➡ K_{edu} **says** (K_{cu} **speaksfor** ϵ/edu/cu)
K_{edu}-⟨ϵ/edu/mit : K_{mit}⟩
…

…
K_{cu}-⟨ϵ/edu/cu/cs : K_{cs}⟩ ➡ K_{cu} **says** (K_{cs} **speaksfor** ϵ/edu/cu/cs)
K_{cu}-⟨ϵ/edu/cu/ece : K_{ece}⟩
…

…
K_{cs}-⟨ϵ/edu/cu/cs/fbs : K_{fbs}⟩ ➡ K_{cs} **says** (K_{fbs} **speaksfor** ϵ/edu/cu/cs/fbs)
K_{cs}-⟨ϵ/edu/cu/ece/la : K_{la}⟩
…

Sample Derivation (1)

K_{fbs} **speaksfor** ϵ/edu/cu/cs/fbs

Sample Derivation (2)

K_{cs} **says** K_{fbs} **speaksfor** ϵ/edu/cu/cs/fbs

K_{CS} **speaksfor** ϵ/edu/cu/cs

ϵ/edu/cu/cs **says** K_{fbs} **speaksfor** ϵ/edu/cu/cs/fbs

ϵ/edu/cu/cs **speaksfor** ϵ/edu/cu/cs/fbs

ϵ/edu/cu/cs/fbs **says** K_{fbs} **speaksfor** ϵ/edu/cu/cs/fbs

K_{fbs} **speaksfor** ϵ/edu/cu/cs/fbs

Sample Derivation (3)

K_{CS} **speaksfor** ϵ/edu/cu/cs

K_{cs} **says** K_{fbs} **speaksfor** ϵ/edu/cu/cs/fbs

~~K_{CS} **speaksfor** ϵ/edu/cu/cs~~

ϵ/edu/cu/cs **says** K_{fbs} **speaksfor** ϵ/edu/cu/cs/fbs

ϵ/edu/cu/cs **speaksfor** ϵ/edu/cu/cs/fbs

ϵ/edu/cu/cs/fbs **says** K_{fbs} **speaksfor** ϵ/edu/cu/cs/fbs

K_{fbs} **speaksfor** ϵ/edu/cu/cs/fbs

Sample Derivation (4)

K_{cu} **says** K_{cs} **speaksfor** ϵ/edu/cu/cs

K_{cu} **speaksfor** ϵ/edu/cu

ϵ/edu/cu **says** K_{cs} **speaksfor** ϵ/edu/cu/cs

ϵ/edu/cu **speaksfor** ϵ/edu/cu/cs

ϵ/edu/cu/cs **says** K_{cs} **speaksfor** ϵ/edu/cu/cs

K_{CS} **speaksfor** ϵ/edu/cu/cs

K_{cs} **says** K_{fbs} **speaksfor** ϵ/edu/cu/cs/fbs

~~K_{CS} **speaksfor** ϵ/edu/cu/cs~~

ϵ/edu/cu/cs **says** K_{fbs} **speaksfor** ϵ/edu/cu/cs/fbs

ϵ/edu/cu/cs **speaksfor** ϵ/edu/cu/cs/fbs

ϵ/edu/cu/cs/fbs **says** K_{fbs} **speaksfor** ϵ/edu/cu/cs/fbs

K_{fbs} **speaksfor** ϵ/edu/cu/cs/fbs

Sample Derivation (5)

K_I **speaksfor** ϵ …

K_{cu} **says** K_{cs} **speaksfor** ϵ/edu/cu/cs

~~K_{cu} **speaksfor** ϵ/edu/cu~~

ϵ/edu/cu **says** K_{cs} **speaksfor** ϵ/edu/cu/cs

ϵ/edu/cu **speaksfor** ϵ/edu/cu/cs

ϵ/edu/cu/cs **says** K_{cs} **speaksfor** ϵ/edu/cu/cs

K_{CS} **speaksfor** ϵ/edu/cu/cs

K_{cs} **says** K_{fbs} **speaksfor** ϵ/edu/cu/cs/fbs

~~K_{CS} **speaksfor** ϵ/edu/cu/cs~~

ϵ/edu/cu/cs **says** K_{fbs} **speaksfor** ϵ/edu/cu/cs/fbs

ϵ/edu/cu/cs **speaksfor** ϵ/edu/cu/cs/fbs

ϵ/edu/cu/cs/fbs **says** K_{fbs} **speaksfor** ϵ/edu/cu/cs/fbs

K_{fbs} **speaksfor** ϵ/edu/cu/cs/fbs

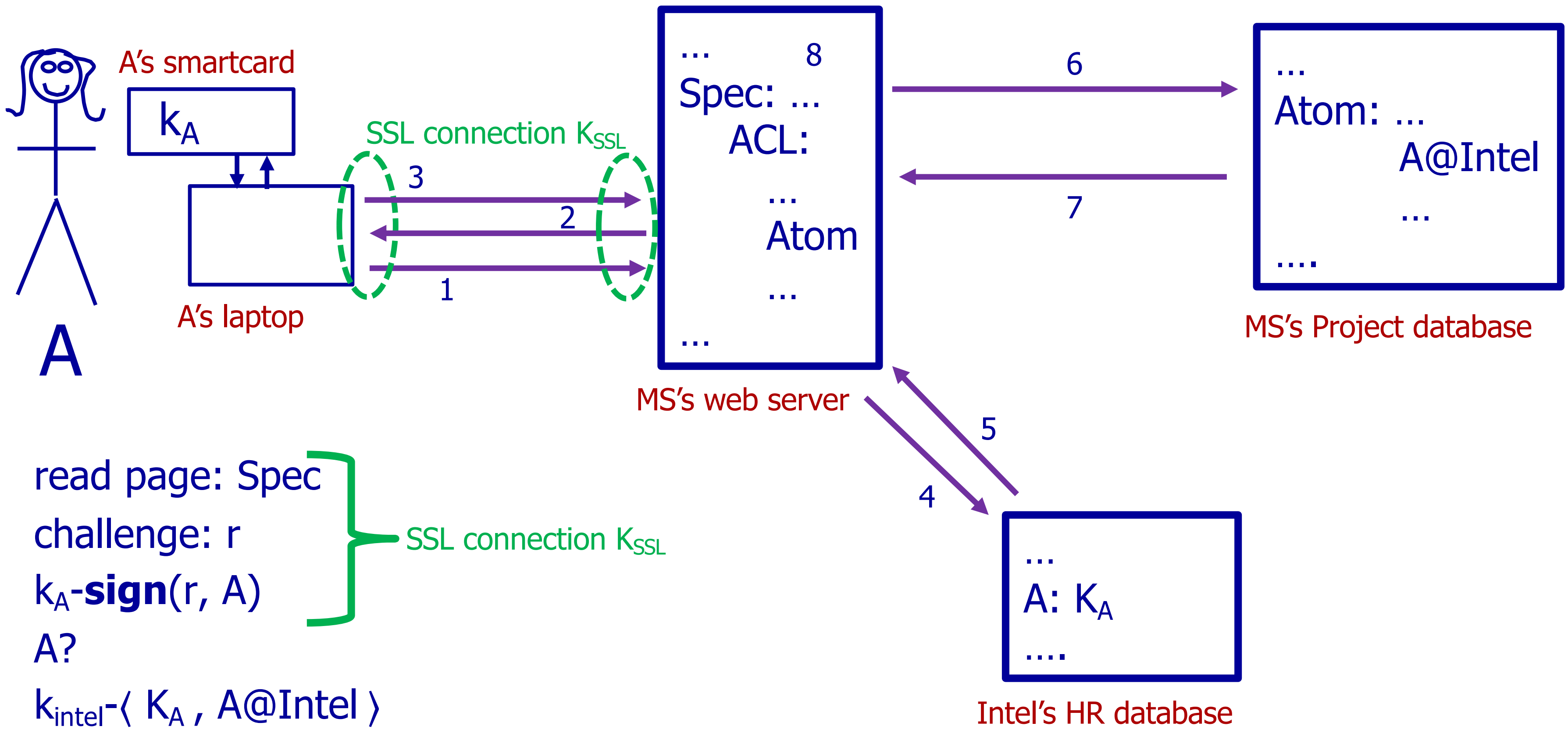
Application 2:
Access to a Joint Project

- A works for Intel and is known as A@Intel.
  - Public key K_A; private key k_A
  - Laptop
  - Member of Atom group
- MS has web page Spec
  - ACL allows access to Spec for members of Atom
  - CAL models as: Atom **speaksfor** Spec
    - Therefore: Atom **says** (access Spec) $\vdash$ Spec **says** (access Spec)

Suppose A requests access a Spec web page…

Application:

Accessing a Joint Project



1. read page: Spec
2. challenge: r
3. k_A-**sign**(r, A)
4. A?
5. k_{intel}-⟨ K_A , A@Intel ⟩
6. A@Intel in Atom?
7. k_{MS}-⟨ A@Intel, Atom ⟩
8. MS web server authorizes access by Atom: Atom ∈ Spec.ACL

(Items 1–3: SSL connection K_{SSL})

CAL Model for Spec Access

1. K_{SSL} **says** (A@Intel **says** (read page: Spec))
2. K_{SSL} **says** r
3. K_{SSL} **says** (K_A **says** (r,A))

    K_{SSL} **speaksfor** K_A since K_A is a subprincipal of K_{SSL}

    Conclude: K_A **says** (r,A)

5. K_{intel} **says** K_A **speaksfor** A@Intel

    K_{intel} **speaksfor** *@Intel, so: K_{intel} **speaksfor** A@Intel

    Conclude: K_A **speaksfor** A@Intel

7. K_{MS} **says** (A@Intel **speaksfor** Atom)

    MS **speaksfor** Atom since Atom is a subprincipal of MS

    K_{MS} **speaksfor** MS defn of K_{MS}

    Conclude: A@Intel **speaksfor** Atom

CAL Model for Spec Access

1. K_{SSL} **says** (A@Intel **says** (read page: Spec))
2. K_{SSL} **says** r
3. K_{SSL} **says** (K_A **says** (r,A))

   K_{SSL} **speaksfor** K_A since K_A is a subprincipal of K_{SSL}

   Conclude: K_A **says** (r,A)

5. K_{intel} **says** K_A **speaksfor** A@Intel

   K_{intel} **speaksfor** *@Intel, so: K_{intel} **speaksfor** A@Intel

   Conclude: K_A **speaksfor** A@Intel

7. K_{MS} **says** (A@Intel **speaksfor** Atom)

   MS **speaksfor** Atom since Atom is a subprincipal of MS

   K_{MS} **speaksfor** MS defn of K_{MS}

   Conclude: A@Intel **speaksfor** Atom

A@Intel **says** (read page: Spec)

CAL Model for Spec Access

1. K_{SSL} **says** (A@Intel **says** (read page: Spec))
2. K_{SSL} **says** r
3. K_{SSL} **says** (K_A **says** (r,A))

   K_{SSL} **speaksfor** K_A since K_A is a subprincipal of K_{SSL}

   Conclude: K_A **says** (r,A)

5. K_{intel} **says** K_A **speaksfor** A@Intel

   K_{intel} **speaksfor** *@Intel, so: K_{intel} **speaksfor** A@Intel

   Conclude: K_A **speaksfor** A@Intel

7. K_{MS} **says** (A@Intel **speaksfor** Atom)

   MS **speaksfor** Atom since Atom is a subprincipal of MS

   K_{MS} **speaksfor** MS defn of K_{MS}

   Conclude: A@Intel **speaksfor** Atom

A@Intel **says** (read page: Spec)

A@Intel **speaksfor** Atom

Access Authorization

A@Intel **says** (read page: Spec)

A@Intel **speaksfor** Atom

Atom **speaksfor** Spec due to Atom $\in$ Spec.ACL

$\vdash$

Spec **says** (read page: Spec)

Application 3:
Protocol 1 for Remote Attestation

Assumptions:

A1: R trusts S and has K_S **speaksfor** S.

A2: S is exec environment for P.

A3: S implements a gating function [k_P-**sign**].

1. R → S: ⟨r, P⟩, where r is fresh nonce
2. S: Generate K_P/k_p where Config([k_P-**sign**]) = {P}
3. S → R: [k_S-**sign**](r, P, K_P)
4. R: Accept K_P provided:
   - Msg 3 verified as from S (by using K_S) and N(D_P)=P holds.

Gating Functions in CAL

$$\frac{\{T\} = \ \mathrm{Config}(\,[k_T - \mathbf{sign}])}{K_T\ \mathbf{speaksfor}\ T}$$

T might be $N(\,P\,)$

Protocol 1: Analysis

1. (3) S → R: [k_S-**sign**](r, P, K_P)
   - K_S **says** (S.r **says** (K_P **speaksfor** P))
2. S.r implements S
   - S.r **speaksfor** S
3. Assumption A1 and CAL Gating Functions Inference Rule
   - K_S **speaksfor** S
4. CAL with 1,3; then 2: S **says** (S **says** (K_P **speaksfor** P))
5. CAL with 4: S **says** (K_P **speaksfor** P)
6. P is a subprincipal of S (since S is exec env for P):
   - S **speaksfor** P
7. CAL with 5, 4: P **says** (K_P **speaksfor** P)
8. CAL Handoff with 7: K_P **speaksfor** P

Review

- Why formalize? Applicability of Authentication Logics.
- Logic refresher (with apologies)
  - Formulas, Theorems, Interpretations, …
- CAL
  - Formulas
  - Interpretations
  - Compound Principals
- Accountability
- Credentials and certificates
- Applications