Flow-Limited Authorization

Owen Arden, Jed Liu, and Andrew Myers
Cornell University
Authorization

Who has **permission** to perform an action?

Who **trusts** whom?

Expressed with **access control lists**, **delegation** between principals

**Respect my Authority!**
Information flow control

Confidentiality:
Who may learn secret values?

Integrity:
Who may influence trusted values?

Expressed with information flow policies (intended flows between principals).
Interactions

Authorization requires integrity

Trust may be confidential

In distributed systems, authorization often requires communication.

How can we reason about who we can communicate with?
Vulnerabilities
Vulnerabilities

ACME has a trade secret:
Vulnerabilities

ACME has a trade secret:

Alice and Bob can learn it.
Vulnerabilities

**ACME** has a trade secret:

Alice and Bob can learn it.

But not Dr. Evil
Delegation loophole

Delegations can enable unauthorized flows.

Can Dr. Evil learn Acme’s secret?
Delegation loophole

Delegations can enable unauthorized flows.

Can Dr. Evil learn Acme’s secret?

In current systems, yes!

Alice  Bob  Evil
Poaching attack

Relabeling data undermines revocation

Does Acme control when Bob learns secrets?
Poaching attack

Relabeling data undermines revocation

Does Acme control when Bob learns secrets? In current systems, no!
Probing attacks

Results may leak information

Can Dr. Evil make authorization queries?
Probing attacks

Results may leak information

Can Dr. Evil make authorization queries?

Could leak secret information!
Read channels
Queries may leak information

Can Dr. Evil observe authorization queries?
Read channels

Queries may leak information

Can Dr. Evil observe authorization queries?

Could leak secret information!
Related Work

• Delegation loophole [Hicks et al. 05]
• Probing attacks [Becker 12]
• Read channels [Zdancewic et al. 02]
• Information flow control in trust management
  • Rx [Swamy et al. 06]
  • RTI [Bandhakavi et al. 08]
• Access control in authorization
  • Automated trust negotiation [Winsborough et al. 00; Winsborough & Li 04]
  • Context-sensitive authorization [Minami & Kotz 05;06]
Related Work

- Delegation loophole [Hicks et al. 05]
- Probing attacks [Becker 12]
- Read channels [Zdancewic et al. 02]
- Information Glow control in trust management
- Rx [Swamy et al. 06]
- RTI [Bandhakavi et al. 08]
- Access control in authorization
- Automated trust negotiation [Winsborough et al. 00; Winsborough & Li 04]
- Context-sensitive authorization [Minami & Kotz 05;06]

Not robust to attack: malicious principals influence authorization decisions in excess of their authority
FLAM: a unified model

- The Flow-Limited Authorization Model is
  - A novel **principal model** unifying authority and information flow control
  - A logic for making **distributed, decentralized** authorization and information flow decisions

- FLAM exhibits **robust authorization**:
  - Delegation loopholes, read channels, and poaching and probing attacks are eliminated
  - Formalized and verified in Coq
Principal model
Principal model

- Authority (of hosts, users, etc.) is represented with *principals*

- *Primitive* principals:
  
  Alice, Bob, Acme, Evil

- *Conjunctive and disjunctive* principals

  Alice \(\land\) Bob  authority of Alice *and* Bob.

  Alice \(\lor\) Bob  authority of Alice *or* Bob
Authority lattice

Lattice join (upper bound):
  Conjunction: Alice ∧ Bob
Lattice meet (lower bound):
  Disjunction: Alice ∨ Bob
Authority lattice

Bob $\succeq$ Acme  "Bob acts for Acme"

Principals higher in the lattice are more trusted (have greater authority).
An **authority projection** is an attenuation of a principal’s authority.

- $\text{Acme} \rightarrow$ *confidentiality* authority
- $\text{Acme} \leftarrow$ *integrity* authority

All authority is a combination:

- $\text{Acme} \equiv \text{Acme} \rightarrow \land \text{Acme} \leftarrow$
- $\text{Acme} \rightarrow \equiv \text{Acme} \rightarrow \land \bot \leftarrow$
- $\text{Acme} \leftarrow \equiv \bot \rightarrow \land \text{Acme} \leftarrow$

$p\rightarrow = \text{confidentiality}$

$p\leftarrow = \text{integrity}$
Information flow lattice

Orders principals by policy *restrictiveness*:

\[ p \sqsubseteq q \triangleq p^\leftarrow \land q^\rightarrow \geq q^\leftarrow \land p^\rightarrow \]

\[ \text{Acme}^\rightarrow \sqsubseteq \text{Bob}^\rightarrow \triangleq \text{Bob}^\rightarrow \geq \text{Acme}^\rightarrow \]

“Acme’s secret flows to Bob”

\[ \text{Bob}^\leftarrow \sqsubseteq \text{Acme}^\leftarrow \triangleq \text{Bob}^\leftarrow \geq \text{Acme}^\leftarrow \]

“Acme endorses flows from Bob”

Lattice join (upper bound):

\[ \text{Alice} \sqcup \text{Bob} \triangleq (\text{Alice} \land \text{Bob})^\rightarrow \land (\text{Alice} \lor \text{Bob})^\leftarrow \]

Lattice meet (lower bound):

\[ \text{Alice} \sqcap \text{Bob} \triangleq (\text{Alice} \lor \text{Bob})^\rightarrow \land (\text{Alice} \land \text{Bob})^\leftarrow \]
Information flow lattice

\[ p^{\rightarrow} = \text{confidentiality} \]

\[ p^{\leftarrow} = \text{integrity} \]
Information flow lattice

Authority projections help **unify** authority and information flow.
Principals “**righter**” are more **restrictive** (more secret, less integrity).
Authority projections help unify authority and information flow. Principals “righter” are more restrictive (more secret, less integrity).

Bob ≽ Acme
Authority projections help **unify** authority and information flow.

Principals “**righter**” are more **restrictive** (more secret, less integrity).

**Bob** ≽ **Acme**

1. **Bob** → ≽ **Acme** →
2. **Bob** ← ≽ **Acme** ←
Authority projections help unify authority and information flow. Principals “righter” are more restrictive (more secret, less integrity).

**Bob ⪰ Acme**
1. Bob → ⪰ Acme →
2. Bob ← ⪰ Acme ←
3. Acme → ⊑ Bob →
4. Bob ← ⊑ Acme ←
Authority projections enable fine-grained delegation:

\[ \text{Bob} \rightarrow \supseteq \text{Acme} \rightarrow \]

1. \[ \text{Acme} \rightarrow \sqsubseteq \text{Bob} \rightarrow \]

No integrity relationship!

\[ p^\rightarrow = \text{confidentiality} \]
\[ p^\leftarrow = \text{integrity} \]
Authority projections enable fine-grained delegation:

\[ \text{Bob} \rightarrow \supseteq \text{Acme} \rightarrow \]

1. \[ \text{Acme} \rightarrow \sqsubseteq \text{Bob} \rightarrow \]

No integrity relationship!
Information flow lattice

Authority projections enable fine-grained delegation:

Bob \rightarrow \triangleright \ Acme \rightarrow

1. Acme \rightarrow \sqsubseteq \ Bob \rightarrow

No integrity relationship!
FLAM judgments
FLAM judgments

\( \mathcal{H}; n; pc; l \vdash p \equiv q \)

Judgment context:
Characterizes the system configuration and information flow context in which a trust relationship holds for a principal.
FLAM judgments

\[ H; n; pc; l \vdash p \succeq q \]

**Judgment context:**
Characterizes the **system configuration** and **information flow context** in which a trust relationship holds for a principal.
FLAM judgments

Query: “Does this judgment hold?”

Result: “This judgment holds.”

\[ \mathcal{H};n;pc;l \vdash p \succeq q \]

Judgment context:
Characterizes the system configuration and information flow context in which a trust relationship holds for a principal.
Trust configuration

A map from hosts to labeled delegations

\[ H; n; pc; l \vdash p \succeq q \]

\((p \succeq q, l) \in H(n)\)
Trust configuration

A map from hosts to labeled delegations

\[ H; n; pc; l \models p \geq q \]

\((\text{Bob} \rightarrow \triangleright \text{Acme} \rightarrow, \text{Acme} \leftarrow) \in H(\text{Acme})\)

\((\text{Evil} \rightarrow \triangleright \text{Bob} \rightarrow, \text{Bob} \leftarrow) \in H(\text{Bob})\)

\(p \rightarrow = \text{confidentiality}\)

\(p \leftarrow = \text{integrity}\)
Current node

The principal performing the derivation.

\[ \mathcal{H}; n; pc; l \vdash p \geq q \]

Only delegations in \( \mathcal{H}(n) \) are directly accessible, others require *communication*.

\[ p \rightarrow = \text{confidentiality} \]
\[ p \leftarrow = \text{integrity} \]
Query label

The confidentiality & integrity of the query.

\[ H; n; pc; l \vdash p \geq q \]

The label of the program counter of the program issuing the query.

To prevent read channels and poaching

\[ pc \rightarrow \quad pc \leftarrow \]
Derivation label

The confidentiality & integrity of the result.

\[ H; n; pc; l \vdash p \succeq q \]

Restricts which delegations the derivation may depend on.

To prevent delegation loopholes and probing attacks.
Robustness
Robust derivations

A derivation is **robust** when the result cannot be influenced by attackers.

\[ \mathcal{H};c;pc;\text{Acme} \vdash \text{Acme} \rightarrow \sqsubseteq \text{Bob} \rightarrow \]

Want an **end-to-end relationship** between **judgment contexts** and **delegation labels**.
Robust derivations

A derivation is **robust** when the result cannot be influenced by attackers

$$\mathcal{H};c;pc;\text{Acme} \not\vdash \text{Acme} \rightarrow \sqsubseteq \text{Evil} \rightarrow$$

Want an **end-to-end relationship** between delegation **labels** and judgment **contexts**
Robust judgments

Robust judgments (\(\vdash\)) are derived using rules with stricter integrity requirements.

\[\mathcal{H};c;pc;l \vdash p \geq q \approx\]

Access control:
\(p\) may **access** \(q\)’s data

\[\mathcal{H};c;pc;l \models p \geq q \approx\]

Flow control:
May **enforce** \(q\)’s policy using \(p\)’s policy
Principal voice

- The **voice** of a principal is the **integrity** necessary to influence that principal’s trust relationships.

\[
\nabla(p\rightarrow \land q\leftarrow) \triangleq p\leftarrow \land q\leftarrow
\]

- Similar to *readers-to-writers* conversion from *robust downgrading* [Chong & Myers 06]

- \(p \trianglerighteq \nabla(q)\) is analogous to *p speaks for q* in authorization logics [Lampson *et al.* 91]
Robust vs non-robust

\[(\text{Bob} \rightarrow \geq \text{Acme} \rightarrow, \text{Acme} \leftarrow) \in \mathcal{H}(\text{Acme})]\
Robust vs non-robust

$(\text{Bob} \rightarrow \text{Acme}, \text{Acme} \leftarrow) \in \mathcal{H}(\text{Acme})$
Robust vs non-robust

$$(\text{Bob} \rightarrow \rightarrow \text{Acme} \rightarrow, \text{Acme} \leftarrow) \in \mathcal{H}(\text{Acme})$$

Implies Bob may **read** Acme’s secrets:

$\checkmark \mathcal{H};c;pc;\text{Acme} \vdash \text{Acme} \rightarrow \subseteq \text{Bob} \rightarrow$
Robust vs non-robust

\[(\text{Bob} \Rightarrow \text{Acme} \Rightarrow, \text{Acme} \Leftarrow) \in \mathcal{H}(\text{Acme})\]

Implies Bob may **read** Acme’s secrets:

\(\checkmark \; \mathcal{H};c;pc;\text{Acme} \vdash \text{Acme} \Rightarrow \subseteq \text{Bob} \Rightarrow\)

*Does not* imply Bob **speaks for** Acme\(\Rightarrow\):

\(\times \; \mathcal{H};c;pc;\text{Acme} \vdash \text{Bob} \Rightarrow \nabla (\text{Acme} \Rightarrow)\)
Robust vs non-robust

\[(\text{Bob} \Rightarrow \text{Acme}^\rightarrow, \text{Acme}^\leftarrow) \in \mathcal{H}(\text{Acme})\]

Implies Bob may read Acme’s secrets:

\[\checkmark \mathcal{H};c;pc;\text{Acme} \vdash \text{Acme}^\rightarrow \subseteq \text{Bob}^\rightarrow\]

Does not imply Bob speaks for Acme:\n
\[\times \mathcal{H};c;pc;\text{Acme} \vdash \text{Bob} \Rightarrow \bigvee(\text{Acme}^\rightarrow)\]

So Bob may not enforce Acme:\n
\[\times \mathcal{H};c;pc;\text{Acme} \nvdash \text{Acme}^\rightarrow \subseteq \text{Bob}^\rightarrow\]
Robust authorization
Robust authorization

- Very general setting:
  - No well-formedness assumptions about trust configuration ($\mathcal{H}$) or queries
  - Derivations may involve any number of (potentially malicious) hosts.
Robust authorization

Consider the **path** from each delegation to the result across **remote hosts** \( n_0 \ldots n_{k-1} \), represented with a principal:

\[
N = n_0 \lor n_1 \lor \ldots \lor n_{k-1}
\]

This principal is an upper bound on the confidentiality and integrity authority of the delegation.

\[
(\text{Bob} \rightarrow \unsim \text{Acme} \rightarrow, l') \in \mathcal{H}(n_0)
\]

\[
\mathcal{H}; n_0; pc_0; l' \vdash \text{Bob} \rightarrow \unsim \text{Acme} \rightarrow
\]

\[
\mathcal{H}; n_i; pc_i; l_i \vdash p_i \unsim q_i
\]

\[
\mathcal{H}; n_k; pc; l \vdash p_k \unsim q_k
\]

A path in the derivation tree
Robust authorization

Consider the **path** from each delegation to the result across **remote hosts** $n_0...n_{k-1}$, represented with a principal:

$$N = n_0 \lor n_1 \lor ... \lor n_{k-1}$$

The following judgments hold:

- $\mathcal{H}; n_k; pc; l \vdash l' \lor N \subseteq l$
- $\mathcal{H}; n_k; pc; l \vdash N \geq pc \rightarrow \wedge l' \leftarrow$
- $\mathcal{H}; n_k; pc; l \vdash n_k \geq (l' \lor N) \rightarrow$

$$(Bob \rightarrow \geq Acme \rightarrow, l') \in \mathcal{H}(n_0)$$

$\mathcal{H}; n_0; pc_0; l' \leftarrow Bob \rightarrow \geq Acme \rightarrow$

$\mathcal{H}; n_i; pc_i; l_i \leftarrow p_i \geq q_i$

$\mathcal{H}; n_k; pc; l \leftarrow p_k \geq q_k$$
Robust authorization

The following judgments hold:

\[ \mathcal{H};n_k;pc;l \vdash l' \lor N \subseteq l \]
\[ \mathcal{H};n_k;pc;l \vdash N \geq pc \rightarrow \land l' \leftarrow \]
\[ \mathcal{H};n_k;pc;l \vdash n_k \geq (l' \lor N) \rightarrow \]
\[ (Bob \rightarrow \geq Acme \rightarrow, l') \in \mathcal{H}(n_0) \]

\[ N = n_0 \lor n_1 \lor ... \lor n_{k-1} \]
Robust authorization

The following judgments hold:

\[ \mathcal{H}; n_k; pc; l \vdash l' \lor N \subseteq l \]

\[ \mathcal{H}; n_k; pc; l \vdash N \trianglerighteq pc \rightarrow \land l' \leftarrow \]

\[ \mathcal{H}; n_k; pc; l \vdash n_k \trianglerighteq (l' \lor N) \rightarrow \]

\[ N = n_0 \lor n_1 \lor \ldots \lor n_{k-1} \]

\[ (Bob \rightarrow \trianglerighteq Acme \rightarrow, l') \in \mathcal{H}(n_0) \]

\[ \mathcal{H}; n_k; pc; l \vdash p_k \trianglerighteq q_k \]
The following judgments hold:

\[
H; nk; \text{pc}; l \vdash \top \lor (l' \lor N) \\
H; nk; \text{pc}; l \vdash N \equiv (l' \lor N) \\
H; nk; \text{pc}; l \vdash \text{pk} \equiv q_k \\
(\text{Bob} \rightarrow \equiv \text{Acme} \rightarrow (l') \in \mathcal{H}(n_0))
\]

Robust authorization

Secret delegations cannot be revealed by the result.

Malicious delegations cannot corrupt the result.
Robust authorization

**Secret delegations** cannot be revealed by the result.

**Malicious delegations** cannot corrupt the result.

**Remote hosts** cannot observe secret queries or corrupt the result.

The following judgments hold:

\[ \mathcal{H}; n_k; pc;l \vdash l' \lor N \subseteq l \]

\[ \mathcal{H}; n_k; pc;l \vdash N \geq pc \rightarrow \land l' \leftarrow \]

\[ \mathcal{H}; n_k; pc;l \vdash n_k \geq (l' \lor N) \rightarrow \]

\[ N = n_0 \lor n_1 \lor \ldots \lor n_{k-1} \]

\[ (Bob \rightarrow \geq Acme \rightarrow, l') \in \mathcal{H}(n_0) \]

\[ \mathcal{H}; n_k; pc;l \vdash p_k \geq q_k \]
Robust authorization

Secret delegations cannot be revealed by the result.

Malicious delegations cannot corrupt the result.

Remote hosts cannot observe secret queries or corrupt the result.

The current host is trusted to enforce confidentiality of the delegations.

The following judgments hold:

\[ \mathcal{H};n_k;pc;l \vdash l' \lor N \subseteq l \]
\[ \mathcal{H};n_k;pc;l \vdash N \supseteq pc \rightarrow \land l' \leftarrow \]
\[ \mathcal{H};n_k;pc;l \vdash n_k \supseteq (l' \lor N) \rightarrow \]

\[ N = n_0 \lor n_1 \lor ... \lor n_{k-1} \]

\[ (Bob \rightarrow \supseteq Acme \rightarrow, l') \in \mathcal{H}(n_0) \]
\[ \mathcal{H};n_k;pc;l \vdash p_k \supseteq q_k \]
Robust authorization

Secret delegations cannot be revealed by the result.

Malicious delegations cannot corrupt the result.

Remote hosts cannot observe secret queries or corrupt the result.

The current host is trusted to enforce confidentiality of the delegations.

No delegation loopholes
No poaching attacks
No probing attacks
No read channels
Also...

Inference rules for FLAM logic

Mechanized proof of Robust Authorization

http://www.cs.cornell.edu/~owen/ (or TR url in paper)

~11k LOC in Coq

Ownership projections

for modeling owned policies, roles

Factorization lemma

Proof search algorithm
Future work

**PreFab**: a language based on FLAM

Additional *authority projections*:

- Availability [Zheng, Myers 05]
- Referential integrity [Liu, Myers 14]

Generalize logic (i.e. $\mathcal{H};n;pc;l \vdash \varphi$)

Formal semantic/operational model
Conclusion

• FLAM: secure, distributed, and decentralized authorization and information flow decisions

• Principals unify authority, confidentiality, and integrity using authority projections

• A logic that exhibits robust authorization, eliminating delegation loopholes, read channels, and poaching and probing attacks

• Formalized and verified in Coq
Thank you!

- FLAM: **secure, distributed, and decentralized** authorization and information flow decisions
- Principals unify authority, confidentiality, and integrity using *authority projections*
- A logic that exhibits **robust authorization**, eliminating delegation loopholes, read channels, and poaching and probing attacks
- Formalized and verified in Coq