Foundations of Access Control for Secure Storage

Avik Chaudhuri

Computer Science Department, University of California, Santa Cruz
avik@cs.ucsc.edu

Secure communication and secure storage

Formal techniques for secure communication

Secure communication and secure storage

Formal techniques for secure communication

- process calculi, type systems, logics, other foundations
- **rigorous** design/analysis of communication protocols!

Secure communication and secure storage

Formal techniques for secure communication

- process calculi, type systems, logics, other foundations
- **rigorous** design/analysis of communication protocols!

But what about
file systems, operating systems, other distributed systems?

Secure communication and secure storage

Formal techniques for secure communication

- process calculi, type systems, logics, other foundations
- **rigorous** design/analysis of communication protocols!

But what about
file systems, operating systems, other distributed systems?

Formal techniques *for secure storage?*

Secure communication and secure storage

Storage is a form of communication!

- reading/writing a file $\approx$ receiving/sending on a channel

Secure communication and secure storage

Storage is a form of communication!

- reading/writing a file $\approx$ receiving/sending on a channel

Previous work on **asymmetric channels** ($\Leftrightarrow$ files?)

Secure communication and secure storage

Storage is a form of communication!

- reading/writing a file $\approx$ receiving/sending on a channel

Previous work on **asymmetric channels** ($\Leftrightarrow$ files?)

Similar applications of **cryptography**

- secure communication on untrusted channels
  $\approx$ secure storage on untrusted servers

Secure communication and secure storage

Storage is a form of communication!

- reading/writing a file $\approx$ receiving/sending on a channel

Previous work on **asymmetric channels** ($\Leftrightarrow$ files?)

Similar applications of **cryptography**

- secure communication on untrusted channels
  $\approx$ secure storage on untrusted servers

But careful about carrying the analogies too far...

Secure communication and secure storage

Role of access control not fully understood.

Secure communication and secure storage

Role of access control not fully understood.

- secure communication seldom relies on access control...

Secure communication and secure storage

Role of access control not fully understood.

- secure communication seldom relies on access control...
  but access control indispensable for secure storage!

Secure communication and secure storage

Role of access control not fully understood.

- secure communication seldom relies on access control...
  but access control indispensable for secure storage!

Can access control complicate security?

Secure communication and secure storage

Role of access control not fully understood.

- secure communication seldom relies on access control...
  but access control indispensable for secure storage!

Can access control complicate security? How?

Secure communication and secure storage

Role of access control not fully understood.

- secure communication seldom relies on access control...
  but access control indispensable for secure storage!

Can access control complicate security? How?
Can access control improve security?

Secure communication and secure storage

Role of access control not fully understood.

- secure communication seldom relies on access control...
  but access control indispensable for secure storage!

Can access control complicate security? How?
Can access control improve security? How?

Secure communication and secure storage

Role of access control not fully understood.

- secure communication seldom relies on access control...
  but access control indispensable for secure storage!

Can access control complicate security? How?
Can access control improve security? How?

Sophisticated **dynamic** effects

- *runtime checks*
- *runtime controls*

Foundations of access control for secure storage

Foundations of access control for secure storage

- *Precise* security properties of *complex* access controls (cryptographic? distributed? dynamic?)

Foundations of access control for secure storage

- *Precise* security properties of *complex* access controls (cryptographic? distributed? dynamic?)
- *Proof techniques* with access control + static analysis for concrete guarantees like secrecy and integrity

Foundations of access control for secure storage

- *Precise* security properties of *complex* access controls (cryptographic? distributed? dynamic?)
- *Proof techniques* with access control + static analysis for concrete guarantees like secrecy and integrity

Complementary lines of work

Foundations of access control for secure storage

- *Precise* security properties of *complex* access controls (cryptographic? distributed? dynamic?)
- *Proof techniques* with access control + static analysis for concrete guarantees like secrecy and integrity

Complementary lines of work

- focus on **correctness** proofs for implementations of access control

Foundations of access control for secure storage

- *Precise* security properties of *complex* access controls (cryptographic? distributed? dynamic?)
- *Proof techniques* with access control + static analysis for concrete guarantees like secrecy and integrity

Complementary lines of work

- focus on **correctness** proofs for implementations of access control
- exploit correct "black-box" access control in proofs of **end-to-end security** properties

Automated analysis techniques derived from logic

Protocol for secure file sharing on untrusted storage

The Plutus file system

Protocol for secure file sharing on untrusted storage

The Plutus file system

Contents secured cryptographically.

Protocol for secure file sharing on untrusted storage

The Plutus file system

Contents secured cryptographically.

Keys for reading/writing contents generated/distributed by owner.

Protocol for secure file sharing on untrusted storage

The Plutus file system

Contents secured cryptographically.

Keys for reading/writing contents
generated/distributed by owner.

Write-key used to encrypt/sign contents.
Read-key used to verify/decrypt contents.

Protocol for secure file sharing on untrusted storage

The Plutus file system

Contents secured cryptographically.

Keys for reading/writing contents
generated/distributed by owner.

Write-key used to encrypt/sign contents.
Read-key used to verify/decrypt contents.

Keys can be revoked by owner (**dynamic access control**)
New keys generated/distributed appropriately.

Schemes for "efficient" dynamic access control

Schemes for "efficient" dynamic access control

New write-key used to write *new* contents.

(Old contents not immediately secured with new write-key.)

Schemes for "efficient" dynamic access control

New write-key used to write *new* contents.
(Old contents not immediately secured with new write-key.)

- Old read-key used to read old contents.

Schemes for "efficient" dynamic access control

New write-key used to write *new* contents.
(Old contents not immediately secured with new write-key.)

- Old read-key used to read old contents.

"**Lazy revocation**"

Schemes for "efficient" dynamic access control

New write-key used to write *new* contents.
(Old contents not immediately secured with new write-key.)

- Old read-key used to read old contents.

"**Lazy revocation**"

New readers *derive* old read-key from new read-key.

Schemes for "efficient" dynamic access control

New write-key used to write *new* contents.
(Old contents not immediately secured with new write-key.)

- Old read-key used to read old contents.

"**Lazy revocation**"

New readers *derive* old read-key from new read-key.
"**Key rotation**"

Schemes for "efficient" dynamic access control

New write-key used to write *new* contents.
(Old contents not immediately secured with new write-key.)

- Old read-key used to read old contents.

"**Lazy revocation**"

New readers *derive* old read-key from new read-key.
"**Key rotation**"

Implementation relies heavily on RSA tricks.

Automated security analysis of Plutus in ProVerif[1]

Applied pi calculus $\longrightarrow$ Horn logic $\circlearrowleft$ **Resolution**

[1] [with Bruno Blanchet, S&P 2008]

Automated security analysis of Plutus in ProVerif[1]

Applied pi calculus $\longrightarrow$ Horn logic $\circlearrowleft$ **Resolution**

Expected properties?

[1] [with Bruno Blanchet, S&P 2008]

Automated security analysis of Plutus in ProVerif[1]

Applied pi calculus $\longrightarrow$ Horn logic $\circlearrowleft$ **Resolution**

Expected properties? *Who knows?*

[1] [with Bruno Blanchet, S&P 2008]

Automated security analysis of Plutus in ProVerif[1]

Applied pi calculus $\longrightarrow$ Horn logic $\circlearrowleft$ **Resolution**

Expected properties? *Specify what seems reasonable...*

[1] [with Bruno Blanchet, S&P 2008]

Automated security analysis of Plutus in ProVerif[1]

Applied pi calculus $\longrightarrow$ Horn logic $\circlearrowleft$ **Resolution**

Expected properties? *Specify what seems reasonable...*

- Weaker secrecy than claimed...

[1] [with Bruno Blanchet, S&P 2008]

Automated security analysis of Plutus in ProVerif[1]

Applied pi calculus $\longrightarrow$ Horn logic $\circlearrowleft$ **Resolution**

Expected properties? *Specify what seems reasonable...*

- Weaker secrecy than claimed...
  - *Writers can act for readers!* + *No "forward" secrecy!*

[1] [with Bruno Blanchet, S&P 2008]

Automated security analysis of Plutus in ProVerif[1]

Applied pi calculus $\longrightarrow$ Horn logic $\circlearrowleft$ **Resolution**

Expected properties? *Specify what seems reasonable...*

- Weaker secrecy than claimed...
  - *Writers can act for readers!* + *No "forward" secrecy!*
- Dangerous attack on integrity...
  - *Adversary can collude with readers to become writers!*

[1] [with Bruno Blanchet, S&P 2008]

Automated security analysis of Plutus in ProVerif[1]

Applied pi calculus $\longrightarrow$ Horn logic $\circlearrowleft$ **Resolution**

Expected properties? *Specify what seems reasonable...*

- Weaker secrecy than claimed...
  - *Writers can act for readers!* + *No "forward" secrecy!*
- Dangerous attack on integrity...
  - *Adversary can collude with readers to become writers!*
  - Inconspicuous mistake, simple fix...

[1] [with Bruno Blanchet, S&P 2008]

Automated security analysis of Plutus in ProVerif[1]

Applied pi calculus $\longrightarrow$ Horn logic $\circlearrowleft$ **Resolution**

Expected properties? *Specify what seems reasonable...*

- Weaker secrecy than claimed...
  - *Writers can act for readers!* + *No "forward" secrecy!*
- Dangerous attack on integrity...
  - *Adversary can collude with readers to become writers!*
  - Inconspicuous mistake, simple fix...

Demo!

[1] [with Bruno Blanchet, S&P 2008]

Security labels in operating systems

Processes, objects tagged with security labels.

Security labels in operating systems

Processes, objects tagged with security labels.

Consider a simple **integrity** mechanism:

- `Low` *processes cannot write* `High` *objects*
- `High` *processes cannot execute* `Low` *objects*

Security labels in operating systems

Processes, objects tagged with security labels.

Consider a simple **integrity** mechanism:

- `Low` *processes cannot write* `High` *objects*
- `High` *processes cannot execute* `Low` *objects*

What if a `High` *process wants to run an executable downloaded from the Internet?*

Security labels in operating systems

Processes, objects tagged with security labels.

Consider a simple **integrity** mechanism:

- `Low` *processes cannot write* `High` *objects*
- `High` *processes cannot execute* `Low` *objects*

What if a `High` *process wants to run an executable downloaded from the Internet?*

- Least privilege: spawn a new `Low` process and run!

Security labels in operating systems

Processes, objects tagged with security labels.

Consider a simple **integrity** mechanism:

- `Low` *processes cannot write* `High` *objects*
- `High` *processes cannot execute* `Low` *objects*

What if a `High` *process wants to run an executable downloaded from the Internet?*

- Least privilege: spawn a new `Low` process and run!
- Trust the executable? Endorse as `High` and run!

Dynamic labels

The Windows Vista operating system

Dynamic labels

The Windows Vista operating system

Processes, objects tagged with **dynamic** integrity labels.

Dynamic labels

The Windows Vista operating system

Processes, objects tagged with **dynamic** integrity labels.

Processes can

- create new processes and objects
- read, write, or execute objects
- lower their own labels
- control labels of objects

under suitable **constraints**

Dynamic labels

The Windows Vista operating system

Processes, objects tagged with **dynamic** integrity labels.

Processes can

- create new processes and objects
- read, write, or execute objects
- lower their own labels
- control labels of objects

under suitable **constraints**

Are information-flow attacks possible?

Dynamic labels

The Windows Vista operating system

Processes, objects tagged with **dynamic** integrity labels.

Processes can

- create new processes and objects
- read, write, or execute objects
- lower their own labels
- control labels of objects

under suitable **constraints**

Are information-flow attacks possible?
Can we eliminate them (by static analysis/runtime monitoring)?

Dynamic labels

The Asbestos operating system

Dynamic labels

The Asbestos operating system

Processes tagged with **dynamic** secrecy labels.

Dynamic labels

The Asbestos operating system

Processes tagged with **dynamic** secrecy labels.

Processes can communicate

- under suitable **constraints**
- with suitable **effects**

Dynamic labels

The Asbestos operating system

Processes tagged with **dynamic** secrecy labels.

Processes can communicate

- under suitable **constraints**
- with suitable **effects**

Dynamically isolate processes that carry secrets...

Dynamic labels

The Asbestos operating system

Processes tagged with **dynamic** secrecy labels.

Processes can communicate

- under suitable **constraints**
- with suitable **effects**

Dynamically isolate processes that carry secrets...

Are information-flow attacks possible?

Dynamic labels

The Asbestos operating system

Processes tagged with **dynamic** secrecy labels.

Processes can communicate

- under suitable **constraints**
- with suitable **effects**

Dynamically isolate processes that carry secrets...

Are information-flow attacks possible?
Can we verify applications written on Asbestos?

Automatic analysis with dynamic logic programs[2]

[2] [with Sriram Rajamani *et al.*, CCS 2008]

Automatic analysis with dynamic logic programs[2]

Model constraints and effects as dynamic logic rules.
Specify security properties as dynamic queries.

[2] [with Sriram Rajamani *et al.*, CCS 2008]

Automatic analysis with dynamic logic programs[2]

Model constraints and effects as dynamic logic rules.
Specify security properties as dynamic queries.

EON = Datalog + **new** + **next**

[2] [with Sriram Rajamani *et al.*, CCS 2008]

Automatic analysis with dynamic logic programs[2]

Model constraints and effects as dynamic logic rules.
Specify security properties as dynamic queries.

EON = Datalog + **new** + **next**
EON queries = Datalog queries + **sequencing**

[2] [with Sriram Rajamani *et al.*, CCS 2008]

Automatic analysis with dynamic logic programs[2]

Model constraints and effects as dynamic logic rules.
Specify security properties as dynamic queries.

EON = Datalog + **new** + **next**
EON queries = Datalog queries + **sequencing**

Decidable query evaluation!

[2] [with Sriram Rajamani *et al.*, CCS 2008]

Automatic analysis with dynamic logic programs[2]

Model constraints and effects as dynamic logic rules.
Specify security properties as dynamic queries.

EON = Datalog + **new** + **next**
EON queries = Datalog queries + **sequencing**

Decidable query evaluation!
(+ efficient under reasonable assumptions)

[2] [with Sriram Rajamani *et al.*, CCS 2008]

Automatic analysis with dynamic logic programs[2]

Model constraints and effects as dynamic logic rules.
Specify security properties as dynamic queries.

EON = Datalog + **new** + **next**
EON queries = Datalog queries + **sequencing**

Decidable query evaluation!
(+ efficient under reasonable assumptions)

- Analyze key design aspects of Windows Vista, Asbestos.
- Verify architecture of a webserver running on Asbestos!

[2] [with Sriram Rajamani *et al.*, CCS 2008]

Automatic analysis with dynamic logic programs[2]

Model constraints and effects as dynamic logic rules.
Specify security properties as dynamic queries.

EON = Datalog + **new** + **next**
EON queries = Datalog queries + **sequencing**

Decidable query evaluation!
(+ efficient under reasonable assumptions)

- Analyze key design aspects of Windows Vista, Asbestos.
- Verify architecture of a webserver running on Asbestos!

Demo!

[2] [with Sriram Rajamani *et al.*, CCS 2008]

Finer analysis techniques derived from PL

Access control + security types

Access control + security types

System interface as a **language**.

Attacker as any program in this language.

Access control + security types

System interface as a **language**.
Attacker as any program in this language.

Access control in *operational semantics* ("rules of the game")

Access control + security types

System interface as a **language**.
Attacker as any program in this language.

Access control in *operational semantics* ("rules of the game")
Security types in *static semantics* ("strategy")

Access control + security types

System interface as a **language**.
Attacker as any program in this language.

Access control in *operational semantics* ("rules of the game")
Security types in *static semantics* ("strategy")

Access control + security types = **Hybrid security types**

Access control + security types

System interface as a **language**.
Attacker as any program in this language.

Access control in *operational semantics* ("rules of the game")
Security types in *static semantics* ("strategy")

Access control + security types = **Hybrid security types**

Hybrid typechecking

Use dynamic checks where possible/as required to typecheck!

Access control + security types

System interface as a **language**.
Attacker as any program in this language.

Access control in *operational semantics* ("rules of the game")
Security types in *static semantics* ("strategy")

Access control + security types = **Hybrid security types**

Hybrid typechecking

Use dynamic checks where possible/as required to typecheck!

- *Rely on access control for soundness*

Access control + security types

System interface as a **language**.
Attacker as any program in this language.

Access control in *operational semantics* ("rules of the game")
Security types in *static semantics* ("strategy")

Access control + security types = **Hybrid security types**

Hybrid typechecking

Use dynamic checks where possible/as required to typecheck!

- *Rely on access control for soundness*
- *Exploit access control for precision*

Access control + security types

System interface as a **language**.
Attacker as any program in this language.

Access control in *operational semantics* ("rules of the game")
Security types in *static semantics* ("strategy")

Access control + security types = **Hybrid security types**

Hybrid typechecking

Use dynamic checks where possible/as required to typecheck!

- *Rely on access control for soundness*
- *Exploit access control for precision*
- ...Identify redundant access control?

Secrecy by typing and access control[3]

[Communication] *secrecy* $\approx$ restrictions on knowledge.

[3] [with Martin Abadi, CSFW 2006]

Secrecy by typing and access control[3]

[Communication] *secrecy* $\approx$ restrictions on knowledge.

- Channel names *must be as secret as* their messages.

[3] [with Martin Abadi, CSFW 2006]

Secrecy by typing and access control[3]

[Communication] *secrecy* $\approx$ restrictions on knowledge.

- Channel names *must be as secret as* their messages.

[Storage] *secrecy* $\approx$ restrictions on knowledge + **access**.

[3] [with Martin Abadi, CSFW 2006]

Secrecy by typing and access control[3]

[Communication] *secrecy* $\approx$ restrictions on knowledge.

- Channel names *must be as secret as* their messages.

[Storage] *secrecy* $\approx$ restrictions on knowledge + **access**.

- File names *may be less secret than* their contents.

[3] [with Martin Abadi, CSFW 2006]

Secrecy by typing and access control[3]

[Communication] *secrecy* $\approx$ restrictions on knowledge.

- Channel names *must be as secret as* their messages.

[Storage] *secrecy* $\approx$ restrictions on knowledge + **access**.

- File names *may be less secret than* their contents.

File type = Channel type + **access bound**

[3] [with Martin Abadi, CSFW 2006]

Secrecy by typing and access control[3]

[Communication] *secrecy* $\approx$ restrictions on knowledge.

- Channel names *must be as secret as* their messages.

[Storage] *secrecy* $\approx$ restrictions on knowledge + **access**.

- File names *may be less secret than* their contents.

File type = Channel type + **access bound**

Formalized in a pi calculus with *stores* and *groups*

[3] [with Martin Abadi, CSFW 2006]

Secrecy by typing and access control[3]

[Communication] *secrecy* ≈ restrictions on knowledge.

- Channel names *must be as secret as* their messages.

[Storage] *secrecy* ≈ restrictions on knowledge + **access**.

- File names *may be less secret than* their contents.

File type = Channel type + **access bound**

Formalized in a pi calculus with *stores* and *groups*

- Groups in operational semantics + type system

[3] [with Martin Abadi, CSFW 2006]

Secrecy by typing and access control[3]

[Communication] *secrecy* ≈ restrictions on knowledge.

- Channel names *must be as secret as* their messages.

[Storage] *secrecy* ≈ restrictions on knowledge + **access**.

- File names *may be less secret than* their contents.

File type = Channel type + **access bound**

Formalized in a pi calculus with *stores* and *groups*

- Groups in operational semantics + type system
  - as principals, secrecy levels

[3] [with Martin Abadi, CSFW 2006]

Dynamic access control + polymorphic types

The worst case view, while simple, is not precise enough!

Dynamic access control + polymorphic types

The worst case view, while simple, is not precise enough!

- Objects often enforce **dynamic** specifications.

Dynamic access control + polymorphic types

The worst case view, while simple, is not precise enough!

- Objects often enforce **dynamic** specifications.
- Invariants must be preserved despite runtime variations.

Dynamic access control + polymorphic types

The worst case view, while simple, is not precise enough!

- Objects often enforce **dynamic** specifications.
- Invariants must be preserved despite runtime variations.
  - *E.g.*, secrecy despite access variations.

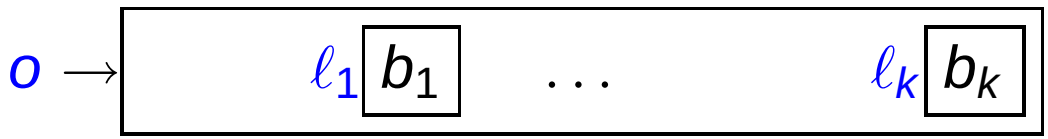
Dynamic access control + polymorphic types

The worst case view, while simple, is not precise enough!

- Objects often enforce **dynamic** specifications.
- Invariants must be preserved despite runtime variations.
  - *E.g.*, secrecy despite access variations.

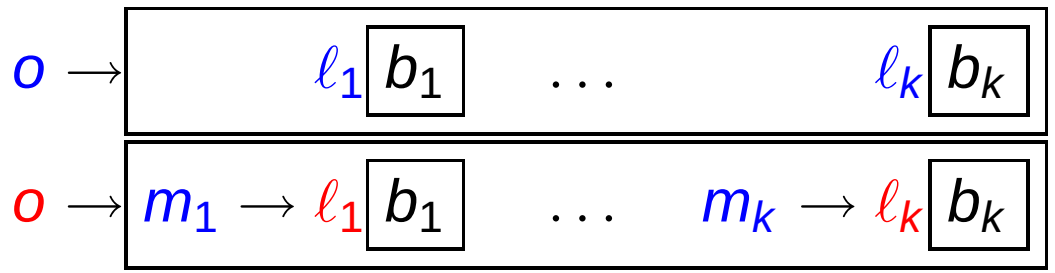
Idea! **Dynamic** access control + **polymorphic** types

Indirections for dynamic access control



conventional object

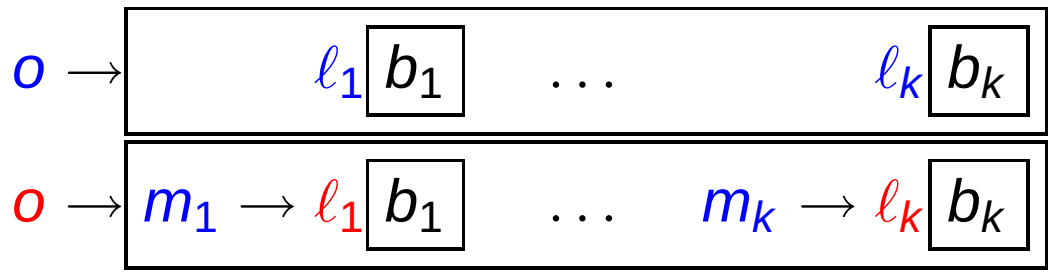
Indirections for dynamic access control



conventional object

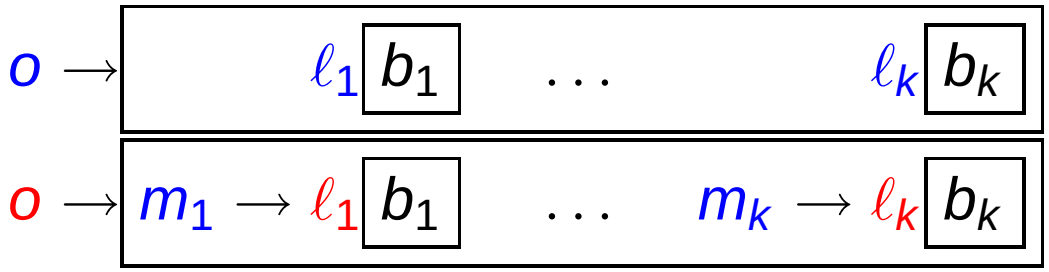
object with *indirections*

Indirections for dynamic access control



Indirections $m_1, \ldots, m_k$ are "temporary aliases"
for methods $\ell_1, \ldots, \ell_k$.

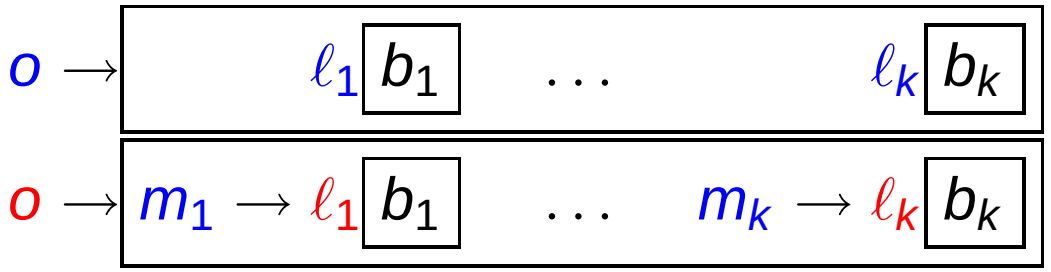
Indirections for dynamic access control



Indirections $m_1, \ldots, m_k$ are "temporary aliases"
for methods $\ell_1, \ldots, \ell_k$.

Knowing (the correct) m_i necessary for calling method ℓ_i.

Indirections for dynamic access control

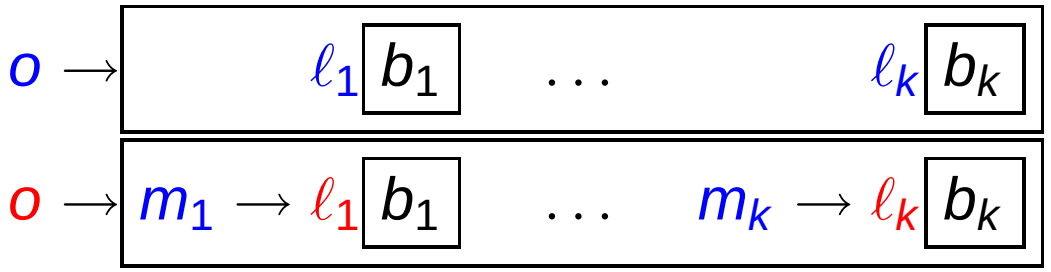


Indirections $m_1, \ldots, m_k$ are "temporary aliases"
for methods $\ell_1, \ldots, \ell_k$.

Knowing (the correct) m_i necessary for calling method ℓ_i.

Dynamic access control enforced via indirections!

Indirections for dynamic access control



Indirections $m_1, \ldots, m_k$ are "temporary aliases"
for methods $\ell_1, \ldots, \ell_k$.

Knowing (the correct) m_i necessary for calling method ℓ_i.

Dynamic access control enforced via indirections!

Knowing o necessary for controlling indirections, methods.

Polymorphic types and dynamic access control[4]

Files

$$f : \mathrm{Obj}[\texttt{read} : X,\ \texttt{write} : X \to 1]$$

[4] [CONCUR 2006]

Polymorphic types and dynamic access control[4]

Files

$$f : \mathrm{Obj}[\texttt{read} : X,\ \texttt{write} : X \to 1]$$

$$r^T : \mathrm{Ind}(T)$$
$$w^T : \mathrm{Ind}(T \to 1)$$

r^T and w^T are indirections for methods at type T.

[4] [CONCUR 2006]

Polymorphic types and dynamic access control[4]

Files

$$f : \mathrm{Obj}[\texttt{read} : X,\ \texttt{write} : X \to 1]$$

$$r^T : \mathrm{Ind}(T)$$
$$w^T : \mathrm{Ind}(T \to 1)$$

r^T and w^T are indirections for methods at type T. **[Plutus]**

[4] [CONCUR 2006]

Polymorphic types and dynamic access control[4]

Files

$$f : \mathrm{Obj}[\texttt{read} : X,\ \texttt{write} : X \to 1]$$

$$r^T : \mathrm{Ind}(T)$$
$$w^T : \mathrm{Ind}(T \to 1)$$

r^T and w^T are indirections for methods at type T. **[Plutus]**

Cryptographic objects

$$s : \mathrm{Obj}[\texttt{key} : X,\ \texttt{encrypt} : Y \to \mathrm{Ind}(X \to Y)]$$

[4] [CONCUR 2006]

Polymorphic types and dynamic access control[4]

Files

$$f : \mathrm{Obj}[\texttt{read} : X,\ \texttt{write} : X \to 1]$$

$$\begin{aligned} r^T &: \mathrm{Ind}(T) \\ w^T &: \mathrm{Ind}(T \to 1) \end{aligned}$$

r^T and w^T are indirections for methods at type T. **[Plutus]**

Cryptographic objects

$$s : \mathrm{Obj}[\texttt{key} : X,\ \texttt{encrypt} : Y \to \mathrm{Ind}(X \to Y)]$$

$$\begin{aligned} k^T &: \mathrm{Ind}(T) \\ e^T &: \mathrm{Ind}(Y \to \mathrm{Ind}(T \to Y)) \\ d^T &: \mathrm{Ind}(T \to S) \end{aligned}$$

[4] [CONCUR 2006]

Dynamic labels and security types[5]

A **calculus** for Windows Vista's security environment.

[5] [with Sriram Rajamani *et al.*, PLAS 2008]

Dynamic labels and security types[5]

A **calculus** for Windows Vista's security environment.

Specify the target security property in the calculus.
(Data-Flow Integrity)

[5] [with Sriram Rajamani *et al.*, PLAS 2008]

Dynamic labels and security types[5]

A **calculus** for Windows Vista's security environment.

Specify the target security property in the calculus.
(Data-Flow Integrity)

Enforce the target security property by (hybrid) typing.

[5] [with Sriram Rajamani *et al.*, PLAS 2008]

Dynamic labels and security types[5]

A **calculus** for Windows Vista's security environment.

Specify the target security property in the calculus.
(Data-Flow Integrity)

Enforce the target security property by (hybrid) typing.
(Access control crucial for sanity.)

[5] [with Sriram Rajamani *et al.*, PLAS 2008]

Certified code?

Typechecking is decidable!

Translate low-level code to our calculus, preserving attacks of interest.
(**Very much future work!**)

Typecheck translated programs.

Data-Flow Integrity (DFI)

Suppose that contents of an object are trusted at some label S.

Data-Flow Integrity (DFI)

Suppose that contents of an object are trusted at some label S.
Then the object never contains data that flows from labels $\sqsubset$ S.

Data-Flow Integrity (DFI)

Suppose that contents of an object are trusted at some label S.
Then the object never contains data that flows from labels $\sqsubset$ S.

Trust is static, declared as an *annotation*.

Data-Flow Integrity (DFI)

Suppose that contents of an object are trusted at some label S.
Then the object never contains data that flows from labels $\sqsubset S$.

Trust is static, declared as an *annotation*.

Operational semantics tracks flows by **explicit substituions**.

Data-Flow Integrity (DFI)

Suppose that contents of an object are trusted at some label S.
Then the object never contains data that flows from labels $\sqsubset$ S.

Trust is static, declared as an *annotation*.

Operational semantics tracks flows by **explicit substituions**.

Type system tracks flows by **effects**.

Distributed access control[6]

The network-attached/object storage protocol

[6] [FCS-ARSPA-WITS 2008; with Martín Abadi, FMSE 2005/FORTE 2006]

Distributed access control[6]

The network-attached/object storage protocol

Can access control be implemented faithfully with **capabilities***?*

[6] [FCS-ARSPA-WITS 2008; with Martín Abadi, FMSE 2005/FORTE 2006]

Distributed access control[6]

The network-attached/object storage protocol

Can access control be implemented faithfully with **capabilities***?*
To what extent?

[6] [FCS-ARSPA-WITS 2008; with Martín Abadi, FMSE 2005/FORTE 2006]

Distributed access control[6]

The network-attached/object storage protocol

Can access control be implemented faithfully with **capabilities***? To what extent?*

Static access control not very problematic.
Dynamic access control presents various problems.

[6] [FCS-ARSPA-WITS 2008; with Martín Abadi, FMSE 2005/FORTE 2006]

Distributed access control[6]

The network-attached/object storage protocol

Can access control be implemented faithfully with **capabilities***? To what extent?*

Static access control not very problematic.
Dynamic access control presents various problems.

Timestamps provide safety, but not full abstraction.

[6] [FCS-ARSPA-WITS 2008; with Martín Abadi, FMSE 2005/FORTE 2006]

Conclusions

Access control plays a sophisticated role in secure storage.

Conclusions

Access control plays a sophisticated role in secure storage.
Dynamic characteristics $\Rightarrow$ pros and cons.

Conclusions

Access control plays a sophisticated role in secure storage.
Dynamic characteristics $\Rightarrow$ pros and cons.

Formal techniques!

Conclusions

Access control plays a sophisticated role in secure storage.
Dynamic characteristics $\Rightarrow$ pros and cons.

Formal techniques!

- Correctness of access control implementations [**logic**]

Conclusions

Access control plays a sophisticated role in secure storage.
Dynamic characteristics $\Rightarrow$ pros and cons.

Formal techniques!

- Correctness of access control implementations [**logic**]
- End-to-end security guarantees with access control [**PL**]

Conclusions

Access control plays a sophisticated role in secure storage.
Dynamic characteristics $\Rightarrow$ pros and cons.

Formal techniques!

- Correctness of access control implementations [**logic**]
- End-to-end security guarantees with access control [**PL**]

Some fresh insights $\Leftarrow$ intersection of security and practice.

Conclusions

Access control plays a sophisticated role in secure storage.
Dynamic characteristics $\Rightarrow$ pros and cons.

Formal techniques!

- Correctness of access control implementations [**logic**]
- End-to-end security guarantees with access control [**PL**]

Some fresh insights $\Leftarrow$ intersection of security and practice.
These insights may apply more generally...