Journey Beyond Full Abstraction:
Exploring Robust Property Preservation for Secure Compilation

https://github.com/secure-compilation/exploring-robust-property-preservation
Good programming languages provide helpful abstractions for writing more secure code
Good programming languages provide helpful abstractions for writing more secure code

- structured control flow, procedures, modules, interfaces, correctness and security specifications, ...
Good programming languages provide helpful abstractions for writing more secure code

- structured control flow, procedures, modules, interfaces, correctness and security specifications, ...

abstractions not enforced when compiling and linking with adversarial low-level code
Good programming languages provide helpful abstractions for writing more secure code

- structured control flow, procedures, modules, interfaces, correctness and security specifications, ...

abstractions not enforced when compiling and linking with adversarial low-level code

- all source-level security guarantees are lost
- linked low-level code can read and write data and code, jump to arbitrary instructions, smash the stack, ...
Secure compilation chains

• Protect source-level abstractions
even against linked adversarial low-level code
Secure compilation chains

• Protect source-level abstractions even against linked adversarial low-level code
  – various enforcement mechanisms possible: processes, SFI, ...
  – shared responsibility: compiler, linker, loader, OS, HW
Secure compilation chains

• Protect source-level abstractions even against linked adversarial low-level code
  – various enforcement mechanisms possible: processes, SFI, ...
  – shared responsibility: compiler, linker, loader, OS, HW

• Enable source-level security reasoning
  – if source program is secure against all source contexts then compiled program is secure against all target contexts
Secure compilation chains

• Protect source-level abstractions even against linked adversarial low-level code
  – various enforcement mechanisms possible: processes, SFI, ...
  – shared responsibility: compiler, linker, loader, OS, HW

• Enable source-level security reasoning
  – if source program is secure against all source contexts then compiled program is secure against all target contexts
  – but what should "is secure" mean?
What properties should we robustly preserve?
What properties should we robustly preserve?

trace properties
(safety & liveness)
What properties should we robustly preserve?

**hyperproperties**

(noninterference)

**trace properties**

(safety & liveness)
What properties should we robustly preserve?

relational hyperproperties
(trace equivalence)

hyperproperties
(noninterference)

trace properties
(safety & liveness)
What properties should we robustly preserve?

relational hyperproperties (trace equivalence)
- Robust Relational Hyperproperty Preservation ($\mathcal{R}r\mathcal{H}P$)
- Robust K-Relational Hyperproperty Preservation ($\mathcal{R}k\mathcal{H}P$)
- Robust 2-Relational Hyperproperty Preservation ($\mathcal{R}2\mathcal{H}P$)
- Robust Hyperproperty Preservation ($\mathcal{H}P$)
- Robust Subset-Closed Hyperproperty Preservation ($\mathcal{SCHC}$)
- Robust K-Subset-Closed Hyperproperty Preservation ($\mathcal{KSCHP}$)
- Robust 2-Subset-Closed Hyperproperty Preservation ($\mathcal{2SCHP}$)
- Robust Hyperproperty Preservation ($\mathcal{H}P$)
- Robust Subproperty Preservation ($\mathcal{SP}$)

hyperproperties (noninterference)
- Robust Relational Hyperproperty Preservation ($\mathcal{R}r\mathcal{H}P$)
- Robust K-Relational Hyperproperty Preservation ($\mathcal{R}k\mathcal{H}P$)
- Robust 2-Relational Hyperproperty Preservation ($\mathcal{R}2\mathcal{H}P$)
- Robust Hyperproperty Preservation ($\mathcal{H}P$)
- Robust Subset-Closed Hyperproperty Preservation ($\mathcal{SCHC}$)
- Robust K-Subset-Closed Hyperproperty Preservation ($\mathcal{KSCHP}$)
- Robust 2-Subset-Closed Hyperproperty Preservation ($\mathcal{2SCHP}$)
- Robust Hyperproperty Preservation ($\mathcal{H}P$)
- Robust Subproperty Preservation ($\mathcal{SP}$)
- Robust Trace Equivalence Preservation ($\mathcal{TEP}$)
- Robust Hypersafety Preservation ($\mathcal{HSP}$)
- Robust K-Hypersafety Preservation ($\mathcal{KHSP}$)
- Robust 2-Hypersafety Preservation ($\mathcal{2HSP}$)

trace properties (safety & liveness)
- Robust Trace Property Preservation ($\mathcal{TP}$)
- Robust Dense Property Preservation ($\mathcal{DP}$)
- Robust Safety Property Preservation ($\mathcal{SP}$)
- Robust Termination-Insensitive Noninterference Preservation ($\mathcal{TINIP}$)

+ determinacy
What properties should we robustly preserve?

Relational hyperproperties (trace equivalence)

Hyperproperties (noninterference)

Trace properties (safety & liveness)

Robust Relational Hyperproperty Preservation (RrHP)
- Robust K-Relational Hyperproperty Preservation (RKrHP)
- Robust 2-Relational Hyperproperty Preservation (R2rHP)

Robust Hyperproperty Preservation (RHP)
- Robust Subset-Closed Hyperproperty Preservation (RSCHC)
- Robust K-Subset-Closed Hyperproperty Preservation (RKSCHP)
- Robust 2-Subset-Closed Hyperproperty Preservation (R2SCHP)

Robust Hyperproperty Preservation (RHP)
- Robust K-Relational Property Preservation (RKrTP)
- Robust 2-Relational Property Preservation (R2rTP)

Robust Relational Trace Equivalence Preservation (RTEP)

Robust Relational Property Preservation (RrTP)
- Robust K-Relational Property Preservation (RKrTP)
- Robust 2-Relational Property Preservation (R2rTP)

Robust Relational XSafety Preservation (RrSP)
- Robust Finite-Relational XSafety Preservation (RFrSC)
- Robust K-Relational XSafety Preservation (RKrSP)
- Robust 2-Relational XSafety Preservation (R2rSP)

Robust Termination-Insensitive Noninterference Preservation (RTINIP)

More secure
More efficient
to enforce
Easier to prove
What properties should we robustly preserve?

- **relational hyperproperties** (trace equivalence)
  - Robust Relational Hyperproperty Preservation ($\mathcal{R}_rHP$)
  - Robust K-Relational Hyperproperty Preservation ($\mathcal{R}_kHP$)
  - Robust 2-Relational Hyperproperty Preservation ($\mathcal{R}_2HP$)

- **hyperproperties** (noninterference)
  - Robust Hyperproperty Preservation ($\mathcal{R}HP$)
  - Robust Subset-Closed Hyperproperty Preservation ($\mathcal{R}_SCCHP$)
  - Robust K-Subset-Closed Hyperproperty Preservation ($\mathcal{R}_KSCHP$)
  - Robust 2-Subset-Closed Hyperproperty Preservation ($\mathcal{R}_2SCHP$)

- **trace properties** (safety & liveness)
  - Robust Trace Property Preservation ($\mathcal{R}TP$)
  - Robust Dense Property Preservation ($\mathcal{R}DP$)

**more secure**

**more efficient**

**Easier to prove**

**only integrity**
What properties should we robustly preserve?

- **relational hyperproperties** (trace equivalence)
  - Robust Relational Hyperproperty Preservation (RrHP)
    - Robust K-Relational Hyperproperty Preservation (RKrHP)
    - Robust 2-Relational Hyperproperty Preservation (R2rHP)
  - Robust Hyperproperty Preservation (RHP)
    - Robust Subset-Closed Hyperproperty Preservation (RSCHC)
    - Robust K-Subset-Closed Hyperproperty Preservation (RKSCHP)
    - Robust 2-Subset-Closed Hyperproperty Preservation (R2SCHP)
  - Robust Trace Property Preservation (RTP)
    - Robust Dense Property Preservation (RDP)
    - Robust Safety Property Preservation (RSP)
    - Robust 2-Hypsafety Property Preservation (R2HSP)

- **hyperproperties** (noninterference)
  - Robust Relational XSafety Preservation (RrSP)
    - Robust Finite-Relational XSafety Preservation (RFrSC)
    - Robust 2-Relational XSafety Preservation (R2rSP)
  - Robust XSafety Preservation (RS)
    - Robust K-XSafety Preservation (RKSP)
    - Robust 2-XSafety Preservation (R2SP)

- **trace properties** (safety & liveness)
  - Robust Termination-Insensitive Noninterference Preservation (RTINIP)

- + data confidentiality
  - only integrity

More secure
More efficient
to enforce
Easier to prove
What properties should we robustly preserve?

**Trace properties** (safety & liveness)
- Only integrity

**Hyperproperties** (noninterference)
- + Data confidentiality
  - Robust Hyperproperty Preservation (RHP)
  - Robust Subset-Closed Hyperproperty Preservation (RSCHC)
  - Robust K-Subset-Closed Hyperproperty Preservation (RKSCHP)
  - Robust 2-Subset-Closed Hyperproperty Preservation (R2SCHP)

  + Code confidentiality
  - Robust Relational Hyperproperty Preservation (RrHP)
  - Robust K-Relational Hyperproperty Preservation (RKrHP)
  - Robust 2-Relational Hyperproperty Preservation (R2rHP)

**More secure**

**More efficient**

Easier to prove
Journey Beyond Full Abstraction
without internal nondeterminism, full abstraction is here
without internal nondeterminism, full abstraction is here
doesn't imply any of our criteria (even assuming compiler correctness)
Journey Beyond Full Abstraction

without internal nondeterminism, full abstraction is here

doesn't imply any of our criteria (even assuming compiler correctness)

no one-size-fits-all criterion!
without internal nondeterminism, full abstraction is here doesn't imply any of our criteria (even assuming compiler correctness) no one-size-fits-all criterion!