CS 599: Formal Methods in Security and Privacy

Noninterference and Relational Hoare Logic

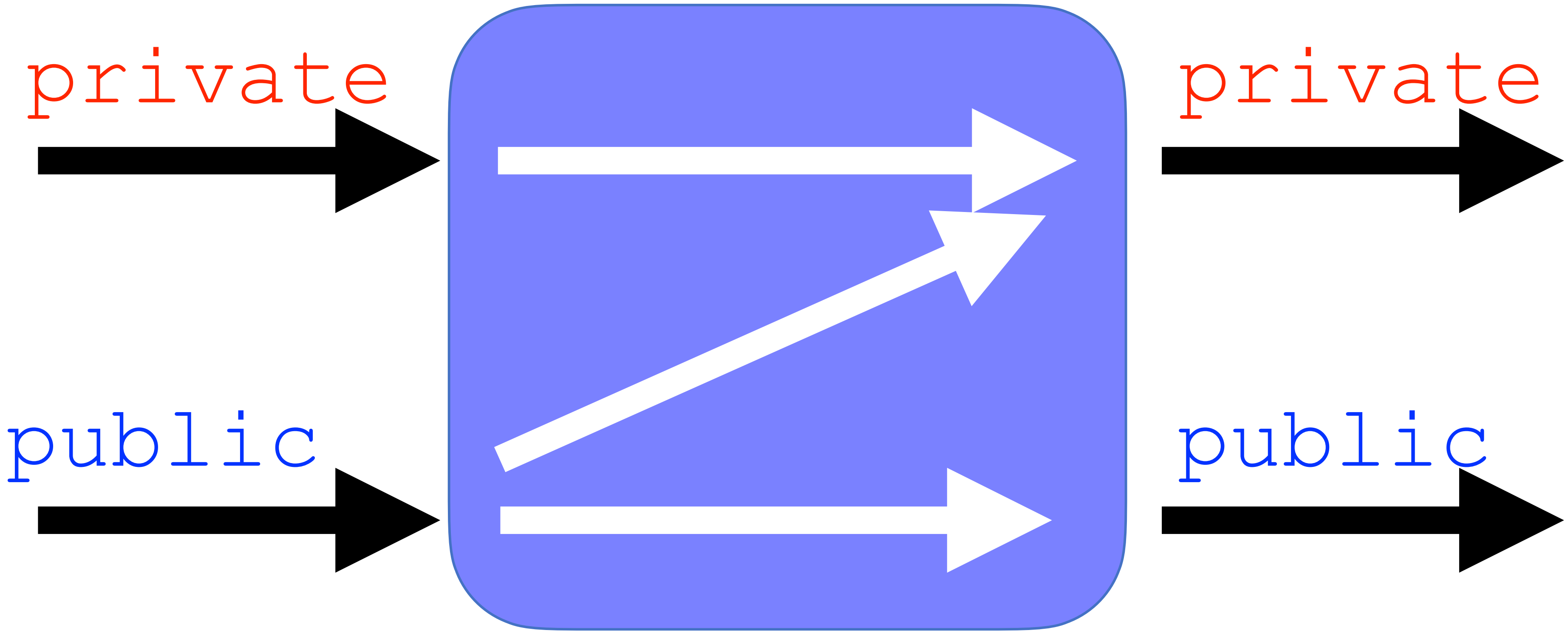
Marco Gaboardi
gaboardi@bu.edu

Alley Stoughton
stough@bu.edu

From the previous classes

Information Flow Control

We want to guarantee that confidential information do not flow in what is considered nonconfidential.

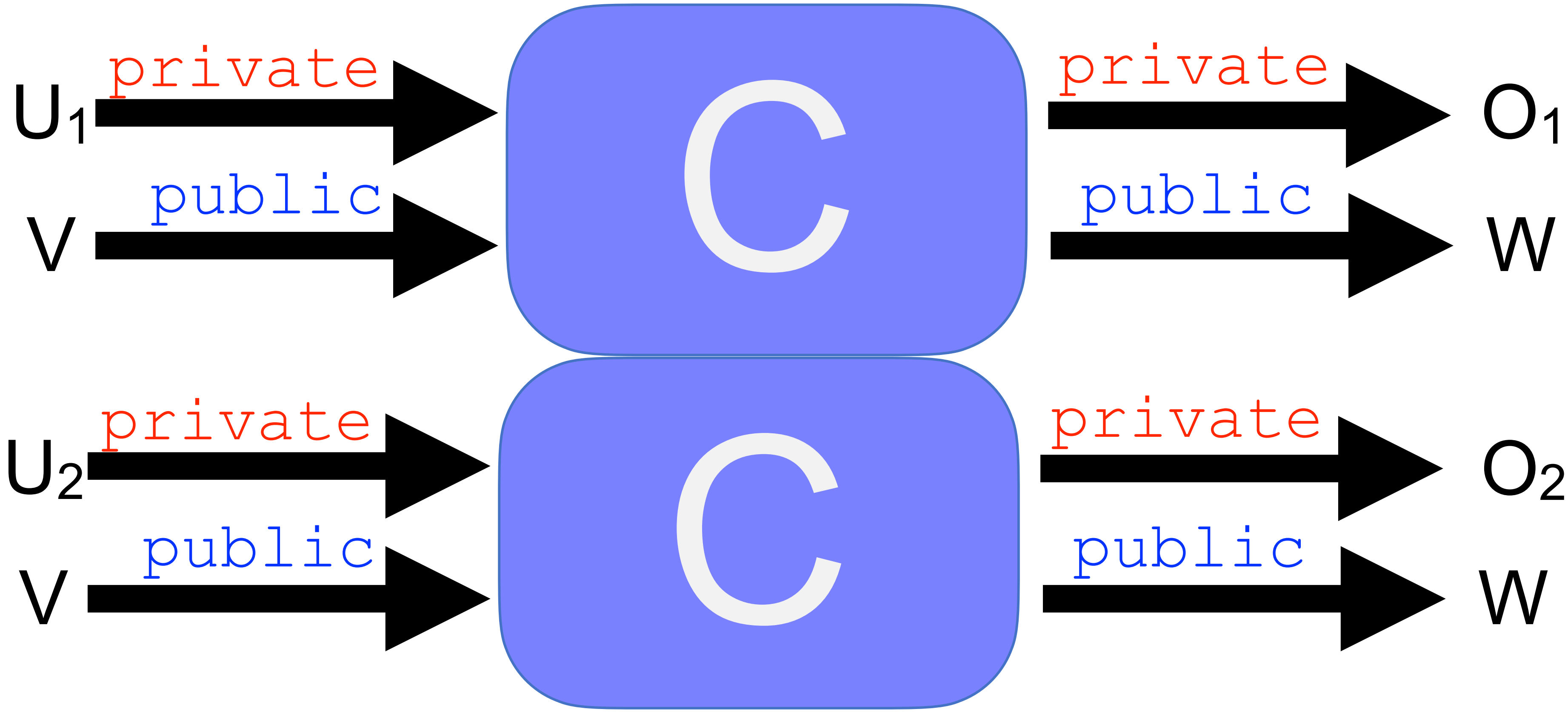


NonInterference

In symbols, c is noninterferent if and only if for every $m_1 \sim_{low} m_2$:

1) $\{c\}_{m1} = \perp$ iff $\{c\}_{m2} = \perp$

2) $\{c\}_{m1} = m_1'$ and $\{c\}_{m2} = m_2'$ implies $m_1' \sim_{low} m_2'$



Relational Hoare Logic - RHL

Precondition
(a logical formula)

Precondition

Program$_1$ ~ Program$_2$

Postcondition

$$c_1 \sim c_2 : P \Rightarrow Q$$

Program

Program

Postcondition
(a logical formula)

Validity of Hoare quadruple

We say that the quadruple $c_1 \sim c_2 : P \Rightarrow Q$ is valid if and only if for every pair of memories m_1, m_2 such that $P(m_1, m_2)$ we have:

1) $\{c_1\}_{m1} = \perp$ iff $\{c_2\}_{m2} = \perp$
2) $\{c_1\}_{m1} = m_1'$ and $\{c_2\}_{m2} = m_2'$ implies $Q(m_1', m_2')$.

Some Rules of Relational Hoare Logic

$$\frac{}{\vdash \mathtt{skip} \sim \mathtt{skip} : P \Rightarrow P}$$

$$\frac{}{\vdash \mathtt{abort} \sim \mathtt{abort} : \mathtt{true} \Rightarrow \mathtt{false}}$$

$$\frac{}{\vdash x_1 := e_1 \sim x_2 := e_2 : P[e_1<1>/x_1<1>, e_2<2>/x_2<2>] \Rightarrow P}$$

$$\frac{\vdash c_1 \sim c_2 : P \Rightarrow R \qquad \vdash c_1' \sim c_2' : R \Rightarrow S}{\vdash c_1 ; c_1' \sim c_2 ; c_2' : P \Rightarrow S}$$

$$\frac{P \Rightarrow S \qquad \vdash c_1 \sim c_2 : S \Rightarrow R \qquad R \Rightarrow Q}{\vdash c_1 \sim c_2 : P \Rightarrow Q}$$

Some Rules of Relational Hoare Logic

$$\frac{\vdash \mathtt{c}_1 \sim \mathtt{c}_2 : \mathtt{e}_1\langle 1\rangle \wedge \mathtt{P} \Rightarrow \mathtt{Q} \qquad \mathtt{P} \Rightarrow \mathtt{e}_1\langle 1\rangle = \mathtt{e}_2\langle 2\rangle \qquad \vdash \mathtt{c}_1' \sim \mathtt{c}_2' : \neg \mathtt{e}_1\langle 1\rangle \wedge \mathtt{P} \Rightarrow \mathtt{Q}}{\vdash \begin{array}{c} \mathtt{if\ e_1\ then\ c_1\ else\ c_1'} \\ \sim \\ \mathtt{if\ e_2\ then\ c_2\ else\ c_2'} \end{array} : \mathtt{P} \Rightarrow \mathtt{Q}}$$

$$\frac{\vdash \mathtt{c}_1 \sim \mathtt{c}_2 : \mathtt{e}_1\langle 1\rangle \wedge \mathtt{P} \Rightarrow \mathtt{P} \qquad \mathtt{P} \Rightarrow \mathtt{e}_1\langle 1\rangle = \mathtt{e2}\langle 2\rangle}{\vdash \begin{array}{c} \mathtt{while\ e_1\ do\ c_1} \\ \sim \\ \mathtt{while\ e_2\ do\ c_2} \end{array} : \mathtt{P} \Rightarrow \mathtt{P} \wedge \neg \mathtt{e}_1\langle 1\rangle}$$

One-sided Rules

$$\frac{\vdash c_1 \sim c_2 : e\langle 1\rangle \wedge P \Rightarrow Q \qquad \vdash c_1' \sim c_2 : \neg e\langle 1\rangle \wedge P \Rightarrow Q}{\vdash \begin{array}{c}\texttt{if } e \texttt{ then } c_1 \texttt{ else } c_1' \\ \sim \\ c_2\end{array} : P \Rightarrow Q}$$

$$\frac{\vdash c_1 \sim c_2 : e\langle 2\rangle \wedge P \Rightarrow Q \qquad \vdash c_1 \sim c_2' : \neg e\langle 2\rangle \wedge P \Rightarrow Q}{\vdash \begin{array}{c} c_1 \\ \sim \\ \texttt{if } e \texttt{ then } c_2 \texttt{ else } c_2'\end{array} : P \Rightarrow Q}$$

Today: More Relational Hoare Logic

Assignment Example

$$\vdash \texttt{x:=x+1} \sim \texttt{y:=y-1}:$$

$$\texttt{x<1>+1=-(y<2>-1)} \Rightarrow \texttt{x<1>=-y<2>}$$

Assignment Example

$$\vdash \texttt{x:=x+1} \sim \texttt{y:=y-1}:$$
$$\texttt{(x<1>=-y<2>)}$$
$$\texttt{[(x+1)<1>/x<1>,(y-1)<2>/y<2>]}$$
$$\Rightarrow$$
$$\texttt{x<1> = -y<2>}$$

Assignment Example

$$
\frac{}{\begin{array}{l}
\vdash x := x+1 \sim y := y-1 : \\
(x\langle 1\rangle = -y\langle 2\rangle) \\
[(x\langle 1\rangle+1)/x\langle 1\rangle, (y\langle 2\rangle-1)/y\langle 2\rangle] \\
\quad \Rightarrow \\
x\langle 1\rangle = -y\langle 2\rangle
\end{array}}
$$

Consequence + Assignment

Example

$$\texttt{x<1>=-y<2>} \Rightarrow \texttt{x<1>+1=-(y<2>-1)}$$

$$\vdash \texttt{x:=x+1} \sim \texttt{y:=y-1:}$$
$$\texttt{x<1>+1=-(y<2>-1)} \Rightarrow \texttt{x<1>=-y<2>}$$

$$\texttt{x<1>=-y<2>} \Rightarrow \texttt{x<1>=-y<2>}$$

$$\vdash \texttt{x:=x+1} \sim \texttt{y:=y-1:}$$
$$\texttt{x<1>=-y<2>} \Rightarrow \texttt{x<1>=-y<2>}$$

How can we prove this?

```
x:private
y:public

if x mod 3 = 0 then
  y:=1
else
  y:=1
```

$: \quad =_{low} \Rightarrow \ =_{low}$

Rules of Relational Hoare Logic If then else - right

$$\frac{\vdash c_1 \sim c_2 : e\langle 2\rangle \wedge P \Rightarrow Q \qquad \vdash c_1 \sim c_2' : \neg e\langle 2\rangle \wedge P \Rightarrow Q}{\vdash c_1 \sim \texttt{if } e \texttt{ then } c_2 \texttt{ else } c_2' : P \Rightarrow Q}$$

How can we prove this?

```
x:public
z:private
y:private

y:=0
z:=0
if x=0 then z:=1;
if z=0 then y:=1
```

$: \quad =_{low} \Rightarrow \; =_{low}$

How can we prove this?

```
s1:public
s2:private
r:private
i:public

proc Compare (s1:list[n] bool,s2:list[n] bool)
i:=0;
r:=0;
while i<n do
 if not(s1[i]=s2[i]) then
    r:=1
 i:=i+1

: n>0 /\ =low ⇒ =low
```

Rules of Relational Hoare-Logic
One-sided Rules

What do we do if our two programs have different forms? There are three pairs of *one-sided* rules.

$$\vdash \begin{array}{c} \texttt{if e then } c_1 \texttt{ else } c_1' \\ \sim \\ c_2 \end{array} : P \Rightarrow Q$$

Rules of Relational Hoare Logic
If-then-else — left

$$\vdash c_1 \sim c_2 : e\langle 1\rangle \wedge P \Rightarrow Q$$

$$\vdash c_1' \sim c_2 : \neg e\langle 1\rangle \wedge P \Rightarrow Q$$

$$\vdash \begin{array}{c} \texttt{if } e \texttt{ then } c_1 \texttt{ else } c_1' \\ \sim \\ c_2 \end{array} : P \Rightarrow Q$$

Rules of Relational Hoare Logic

If-then-else — right

$$\frac{\vdash c_1 \sim c_2 : e\langle 2\rangle \wedge P \Rightarrow Q \qquad \vdash c_1 \sim c_2' : \neg e\langle 2\rangle \wedge P \Rightarrow Q}{\vdash \begin{array}{c} c_1 \\ \sim \\ \texttt{if } e \texttt{ then } c_2 \texttt{ else } c_2' \end{array} : P \Rightarrow Q}$$

Rules of Relational Hoare Logic

Assignment — left

$$\frac{}{\vdash \texttt{x:=e} \sim \texttt{skip}: \texttt{P[e<1>/x<1>]} \Rightarrow \texttt{P}}$$

Assignment — left

$$\frac{}{\vdash \texttt{x:=e} \sim \texttt{skip}:\ \texttt{P[e<1>/x<1>]} \Rightarrow \texttt{P}}$$

Assignment — right

$$\vdash \texttt{skip} \sim \texttt{x:=e:}\ \texttt{P[e<2>/x<2>]} \Rightarrow \texttt{P}$$

Also pair of one-sided rules for while — we'll ignore for now

Rules of Relational Hoare Logic
Program Equivalence Rule

$$\frac{\models P : c_1{}' \equiv c_1 \qquad \models P : c_2{}' \equiv c_2 \qquad \vdash c_1{}' \sim c_2{}' : P \Rightarrow Q}{\vdash c_1 \sim c_2 : P \Rightarrow Q}$$

$\models P : c_1 \equiv c_2$ means $\{c_1\}_m = \{c_2\}_m$ for all m such that $P(m)$

Rules of Relational Hoare Logic
Program Equivalences

$\models$P : skip;c ≡ c

$\models$P : c;skip ≡ c

$\models$P:(c1;c2);c3 ≡ c1;(c2;c3)

...

Rules of Relational Hoare Logic
Combining Composition and Equivalence

We can combine the Composition and Program Equivalence Rules to split commands where we like:

$$\frac{\vdash c_1;c_2 \sim c_1' : P \Rightarrow R \qquad \vdash c_3 \sim c_2';c_3' : R \Rightarrow Q}{\vdash c_1;c_2;c_3 \sim c_1';c_2';c_3' : P \Rightarrow Q}$$

Rules of Relational Hoare Logic

Combining Composition and Equivalence

$$\dfrac{\dfrac{\vdash c_1 \sim \texttt{skip} : P \Rightarrow R \qquad \vdash c_2 \sim c_1' : R \Rightarrow Q}{\vdash c_1;c_2 \sim \texttt{skip};c_1' : P \Rightarrow Q}}{\vdash c_1;c_2 \sim c_1' : P \Rightarrow Q}$$

Rules of Relational Hoare Logic
Combining Composition and Equivalence

$$\vdash c_1 \sim c_1' : P \Rightarrow R \qquad \vdash c_2 \sim \mathtt{skip} : R \Rightarrow Q$$

$$\dfrac{\dfrac{}{\vdash c_1;c_2 \sim c_1';\mathtt{skip} : P \Rightarrow Q}}{\vdash c_1;c_2 \sim c_1' : P \Rightarrow Q}$$

Relational Hoare Logic in EasyCrypt

- EasyCrypt's implementation of Relational Hoare Logic has much in common with its implementation of Hoare Logic.
- Look for the pRHL tactics in Section 3.4 of the EasyCrypt Reference Manual (the "p" stands for "probabilistic", but ignore that for now).

Soundness

If we can derive $\vdash \mathtt{c_1 \sim c_2 : P \Rightarrow Q}$ through the rules of the logic, then the quadruple $\mathtt{c_1 \sim c_2 : P \Rightarrow Q}$ is valid.

Validity of Hoare quadruple

We say that the quadruple $c_1 \sim c_2 : P \Rightarrow Q$ is valid if and only if for every pair of memories m_1, m_2 such that $P(m_1, m_2)$ we have:

1) $\{c_1\}_{m1} = \perp$ iff $\{c_2\}_{m2} = \perp$
2) $\{c_1\}_{m1} = m_1'$ and $\{c_2\}_{m2} = m_2'$ implies $Q(m_1', m_2')$.

How do we check this?

Relative Completeness

If a quadruple $c_1 \sim c_2 : P \Rightarrow Q$ is valid, and we have an oracle to derive all the true statements of the form $P \Rightarrow S$ and of the form $R \Rightarrow Q$, then we can derive $\vdash c_1 \sim c_2 : P \Rightarrow Q$ through the rules of the logic.

Soundness and completeness with respect to Hoare Logic

$$\vdash_{\mathtt{RHL}} \mathtt{c}_1 \sim \mathtt{c}_2 : \mathtt{P} \Rightarrow \mathtt{Q}$$

iff

$$\vdash_{\mathtt{HL}} \mathtt{c}_1 ; \mathtt{c}_2 : \mathtt{P} \Rightarrow \mathtt{Q}$$

Under the assumption that we can partition the memory adequately, and that we have termination.

Possible projects

In Easycrypt

- Look at how to guarantee trace-based noninterference.
- Look at how to guarantee side-channel free noninterference.
- Look at the relations between self-composition and relational logic.

Not related to Easycrypt

- Look at type systems for non-interference.
- Look at other methods for relational reasoning
- Look at declassification