CS 591: Formal Methods in Security and Privacy

Non-interference

Marco Gaboardi
gaboardi@bu.edu

Alley Stoughton
stough@bu.edu

From the previous classes

Hoare triple

Precondition
(a logical formula)

`Precondition`

`Program`

`Postcondition`

$$c : P \Rightarrow Q$$

Program

Postcondition
(a logical formula)

Validity of Hoare triple

We say that the triple `c:P⇒Q` is valid

if and only if

for every memory `m` such that `P(m)` and memory m' such that $\{c\}_m = m'$ we have `Q(m')`.

Is this condition easy to check?

Rules of Hoare Logic
Skip

$$\frac{}{\vdash \texttt{skip: } P \Rightarrow P}$$

Rules of Hoare Logic abort

$$\frac{}{\vdash \texttt{abort: true} \Rightarrow \texttt{false}}$$

Rules of Hoare Logic
Assignment

$$\frac{}{\vdash \texttt{x:=e} \;:\; P[e/x] \Rightarrow P}$$

Rules of Hoare Logic
Composition

$$\frac{\vdash \mathtt{c} : \mathtt{P} \Rightarrow \mathtt{R} \qquad\qquad \vdash \mathtt{c}' : \mathtt{R} \Rightarrow \mathtt{Q}}{\vdash \mathtt{c};\mathtt{c}' : \ \mathtt{P} \Rightarrow \mathtt{Q}}$$

Rules of Hoare Logic
Consequence

$$\frac{P \Rightarrow S \qquad \vdash c : S \Rightarrow R \qquad R \Rightarrow Q}{\vdash c : \; P \Rightarrow Q}$$

We can weaken P, i.e. replace it by something that is implied by P. In this case S.

We can strengthen Q, i.e. replace it by something that implies Q. In this case R.

Rules of Hoare Logic
If then else

$$\frac{\vdash \mathtt{c}_1 : \mathtt{e} \wedge \mathtt{P} \Rightarrow \mathtt{Q} \qquad \vdash \mathtt{c}_2 : \neg \mathtt{e} \wedge \mathtt{P} \Rightarrow \mathtt{Q}}{\vdash \mathtt{if\ e\ then\ c}_1\ \mathtt{else\ c}_2 : \mathtt{P} \Rightarrow \mathtt{Q}}$$

Rules of Hoare Logic
While

$$\frac{\vdash c : e \wedge P \Rightarrow P}{\vdash \texttt{while } e \texttt{ do } c : P \Rightarrow P \wedge \neg e}$$

Invariant

Soundness

If we can derive $\vdash \texttt{c} : \texttt{P} \Rightarrow \texttt{Q}$ through the rules of the logic, then the triple $\texttt{c} : \texttt{P} \Rightarrow \texttt{Q}$ is valid.

Relative Completeness

$$\frac{\mathtt{P} \Rightarrow \mathtt{S} \qquad \vdash \mathtt{c} : \mathtt{S} \Rightarrow \mathtt{R} \qquad \mathtt{R} \Rightarrow \mathtt{Q}}{\vdash \mathtt{c} : \mathtt{P} \Rightarrow \mathtt{Q}}$$

If a triple $\mathtt{c} : \mathtt{P} \Rightarrow \mathtt{Q}$ is valid, and we have an oracle to derive all the true statements of the form $\mathtt{P} \Rightarrow \mathtt{S}$ and of the form $\mathtt{R} \Rightarrow \mathtt{Q}$, then we can derive $\vdash \mathtt{c} : \mathtt{P} \Rightarrow \mathtt{Q}$ through the rules of the logic.

Some Examples of Security Properties

- Access Control
- Encryption
- Malicious Behavior Detection
- Information Filtering
- Information Flow Control

Private vs Public

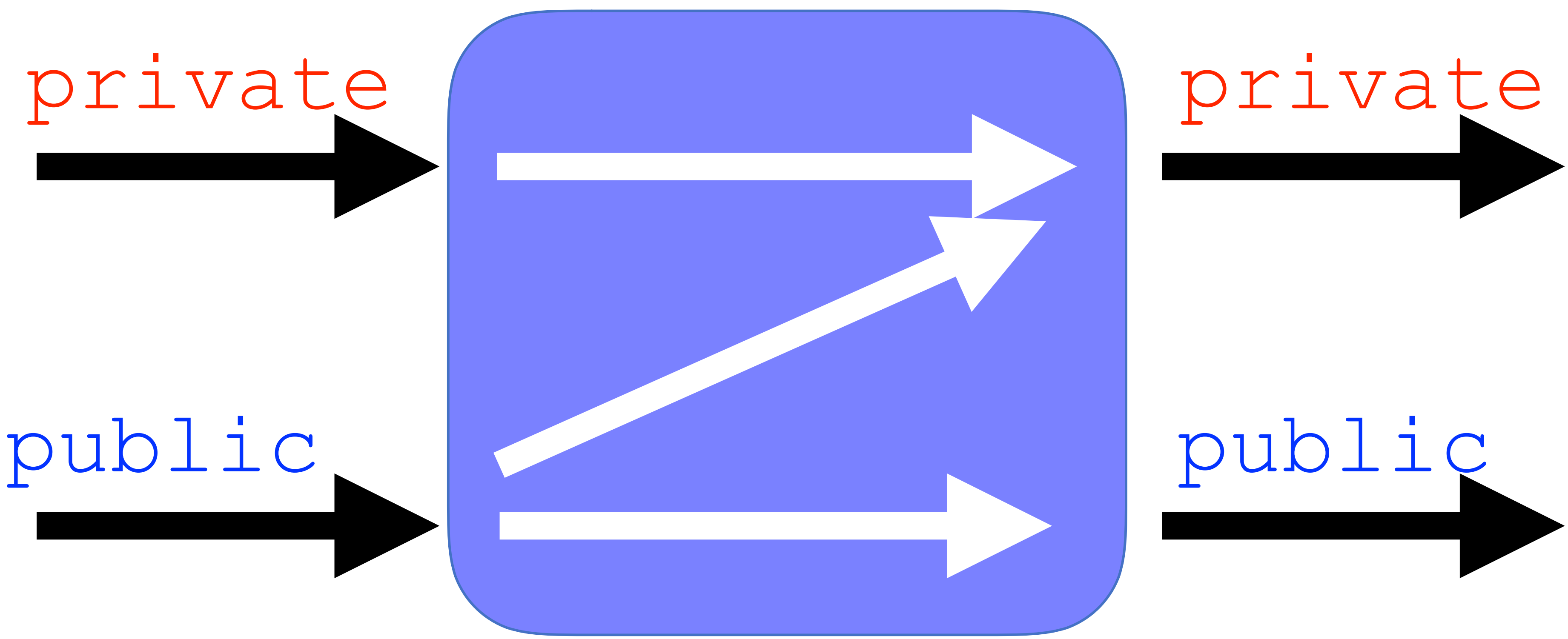
We want to distinguish confidential information that need to be kept secret from nonconfidential information that can be accessed by everyone.

We assume that every variable is tagged with one either public or private.

`x:public` `x:private`

Information Flow Control

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



Today: Noninterference - Relational Hoare Logic

How can we formulate a policy that forbids flows from private to public?

Low equivalence

Two memories m_1 and m_2 are low equivalent if and only if they coincide in the value that they assign to public variables.

In symbols: $m_1 \sim_{low} m_2$

Noninterference

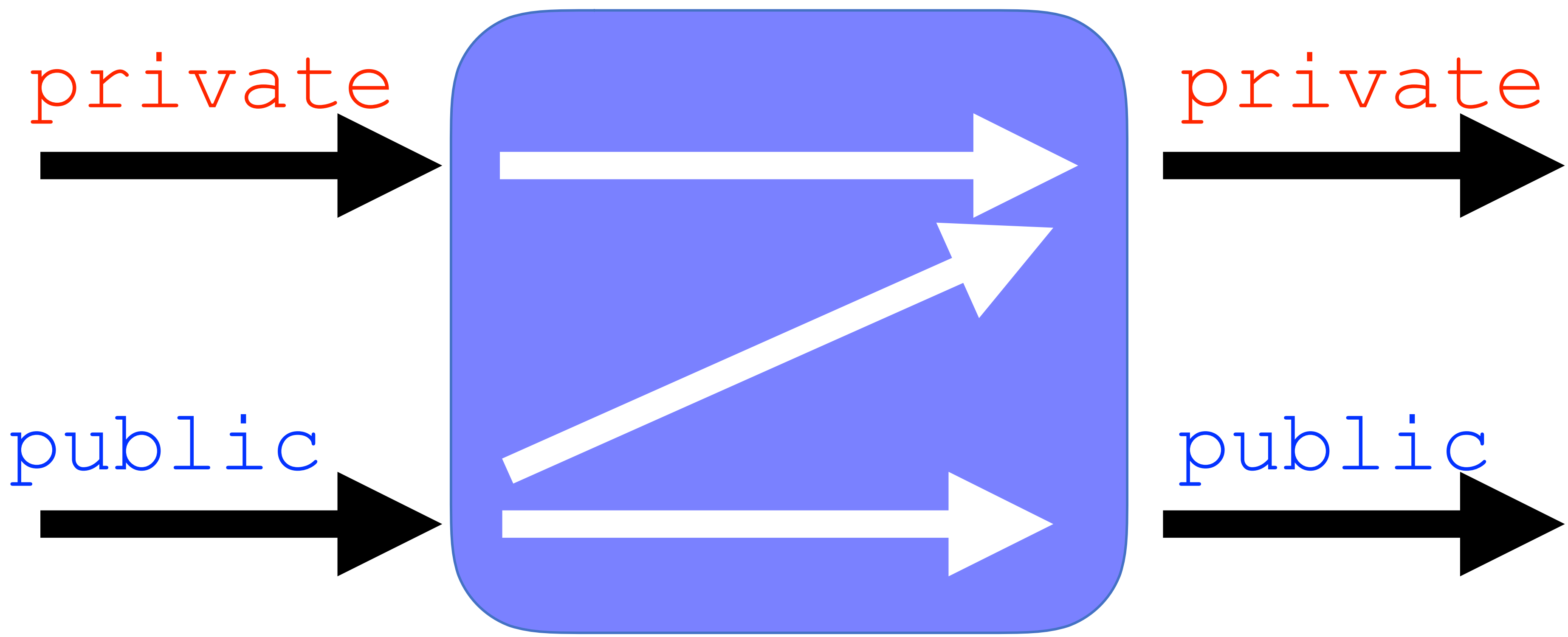
A program `prog` is noninterferent if and only if, whenever we run it on two low equivalent memories m_1 and m_2 we have that:

1) Either both terminate or both non-terminate
2) If they both terminate we obtain two low equivalent memories m_1' and m_2'.

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 $m_1' \sim_{low} m_2'$



Does this program satisfy noninterference?

```
x:private
y:public

x:=y
```

Yes

Does this program satisfy noninterference?

```
x:private
y:public

y:=x
```

No

Is this program secure?

```
x:private
y:public

y:=x
y:=5
```

Yes

Does this program satisfy noninterference?

```
x:private
y:public

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

Yes

Does this program satisfy noninterference?

```
x:private
y:public

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

No

Does this program satisfy noninterference?

```
x:private
y:public

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

Yes

Does this program satisfy noninterference?

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

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

Yes

Does this program satisfy noninterference?

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

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

No

Does this program satisfy noninterference?

```
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 /\ r=0 do
 if not(s1[i]=s2[i]) then
    r:=1
 i:=i+1
```

No

Does this program satisfy noninterference?

```
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
```

Yes

How can we prove our programs noninterferent?

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 $m_1' \sim_{low} m_2'$

Is this condition easy to check?

Can we use the tool we studied so far?

Precondition
(a logical formula)

Precondition

`Program`

Postcondition

$$c : P \Rightarrow Q$$

Program

Postcondition
(a logical formula)

Validity of Hoare triple

We say that the triple $\texttt{c:P}\Rightarrow\texttt{Q}$ is valid

if and only if

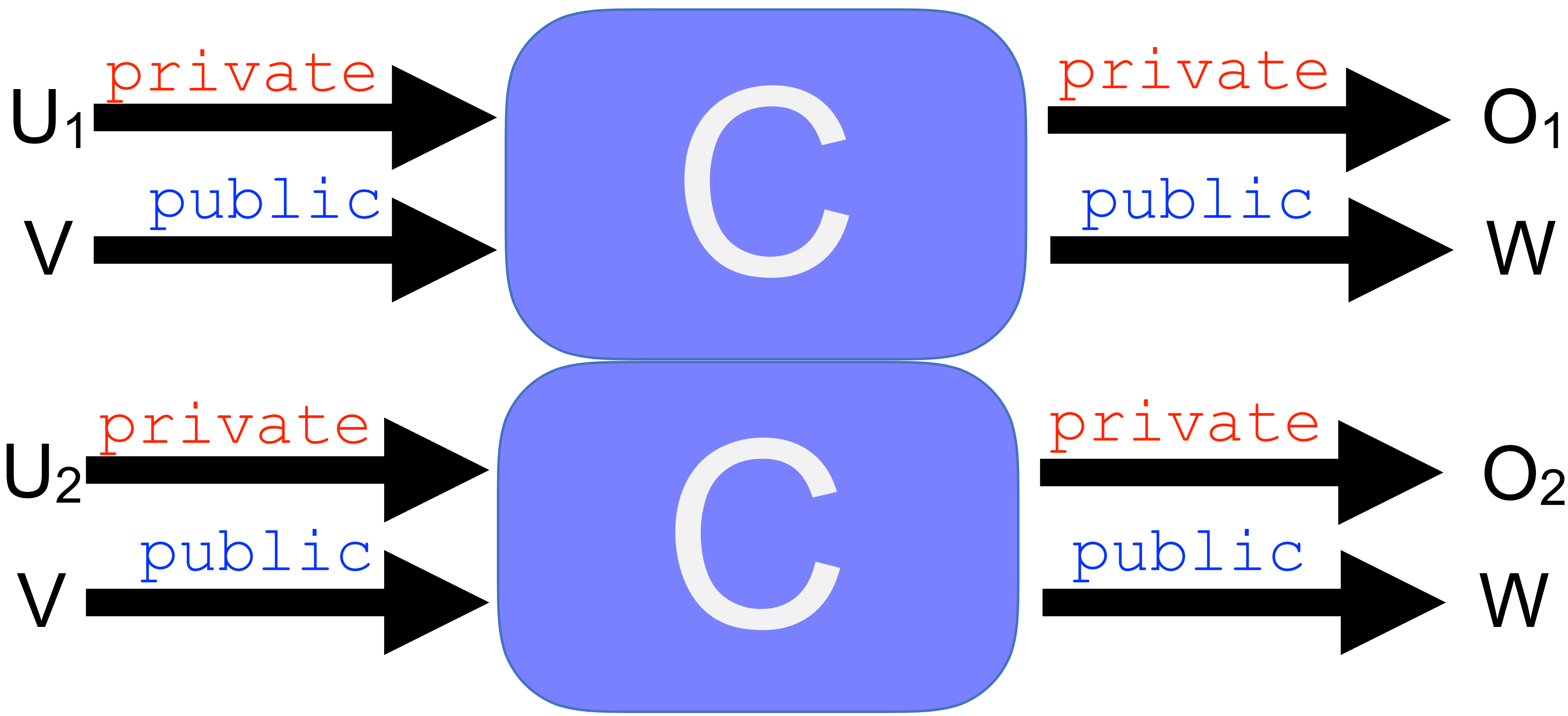
for every memory $\texttt{m}$ such that $\texttt{P(m)}$
and memory m' such that $\{\texttt{c}\}_{\texttt{m}}=\texttt{m'}$
we have $\texttt{Q(m')}$.

Validity talks only about one memory. How can we manage two memories?

Relational Property

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

1) $\{c\}_{m1} = \bot$ iff $\{c\}_{m2} = \bot$
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
$\texttt{Program}_1 \sim \texttt{Program}_2$
Postcondition

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

Program

Program

Postcondition
(a logical formula)

Relational Assertions

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

Need to talk about variables
of the two memories

$$c_1 \sim c_2 : x\langle 1 \rangle \leq x\langle 2 \rangle \Rightarrow x\langle 1 \rangle \geq x\langle 2 \rangle$$

Tags describing which
memory we are referring to.

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 $Q(m_1', m_2')$.

Is this easy to check?

Rules of Relational Hoare Logic
Skip

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

Rules of Relational Hoare Logic
abort

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

Rules of Relational Hoare Logic
Assignment

⊢x:=e~x:=e:

P[e<1>/x<1>,e<2>/x<2>]⇒P