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

\[ c : P \implies Q \]

- **Precondition**
  - (a logical formula)

- **Postcondition**
  - (a logical formula)

- **Program**

---

Precondition

Program

Postcondition
Validity of Hoare triple

We say that the triple $c : P \Rightarrow Q$ 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

\[ \vdash \text{skip: } P \Rightarrow P \]
Rules of Hoare Logic

\[ \vdash \text{abort} : \text{true} \Rightarrow \text{false} \]
Rules of Hoare Logic

Assignment

\[ \vdash x := e : P[e/x] \Rightarrow P \]
Rules of Hoare Logic Composition

\[\vdash c : P \Rightarrow R \quad \vdash c' : R \Rightarrow Q \]

\[\vdash c; c' : P \Rightarrow Q\]
Rules of Hoare Logic

Consequence

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

\[ \vdash c_1 : e \land P \Rightarrow Q \quad \vdash c_2 : \neg e \land P \Rightarrow Q \]

\[ \vdash \text{if } e \text{ then } c_1 \text{ else } c_2 : P \Rightarrow Q \]
Rules of Hoare Logic

While

\[\forall c : e \land P \Rightarrow P\]

\[\forall\text{while } e \text{ do } c : P \Rightarrow P \land \neg e\]

Invariant
If we can derive $\vdash c : P \Rightarrow Q$ through the rules of the logic, then the triple $c : P \Rightarrow Q$ is valid.
Relative Completeness

\[ \vdash c : S \implies R \quad R \implies Q \]

\[ \vdash c : P \implies Q \]

If a triple \( c : P \implies Q \) is valid, and we have an oracle to derive all the true statements of the form \( P \implies S \) and of the form \( R \implies Q \), then we can derive \( \vdash c : P \implies Q \) 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.

\[ x : \text{public} \quad \text{and} \quad x : \text{private} \]
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: $m_1 \sim_{\text{low}} m_2$
A program \texttt{prog} is \textit{noninterferent} if and only if, whenever we run it on two \textit{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 \textit{low equivalent} memories \( m_1' \) and \( m_2' \).
Noninterference

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

1) $\{c\}_{m_1} = \bot$ iff $\{c\}_{m_2} = \bot$

2) $\{c\}_{m_1} = m_1'$ and $\{c\}_{m_2} = m_2'$ implies $m_1' \sim_{\text{low}} m_2'$
Does this program satisfy noninterference?

\[
\begin{array}{l}
x: \text{private} \\
y: \text{public} \\
x := y
\end{array}
\]
Does this program satisfy noninterference?

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

\( y := x \)

No
Is this program secure?

x: private
y: public

\[ y := x \]
\[ y := 5 \]
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?

\[ x: \text{private} \]
\[ y: \text{public} \]

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?

\[
\begin{align*}
x & : \text{public} \\
z & : \text{public} \\
y & : \text{private}
\end{align*}
\]

\[
\begin{align*}
y & := 0 \\
z & := 0 \\
\text{if } x = 0 & \text{ then } z := 1; \\
\text{if } z = 0 & \text{ then } y := 1;
\end{align*}
\]

Yes
Does this program satisfy noninterference?

\[
\begin{align*}
x & : \text{private} \\
z & : \text{public} \\
y & : \text{private} \\
y & : = 0 \\
z & : = 0 \\
\text{if } x = 0 \text{ then } z & : = 1; \\
\text{if } z = 0 \text{ then } y & : = 1; \\
\end{align*}
\]

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 \(\land\) 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
      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 $m_1 \sim_{\text{low}} m_2$:

1) $\{c\}_{m_1} = \bot$ iff $\{c\}_{m_2} = \bot$

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

Is this condition easy to check?
Can we use the tool we studied so far?

Precondition

Program

Postcondition

\[ c : P \implies Q \]

Precondition

(a logical formula)

Program

Postcondition

(a logical formula)
Validity of Hoare triple

We say that the triple $c : P \Rightarrow Q$ is valid if and only if for every memory $m$ such that $P(m)$ and memory $m'$ such that $\{c\}_m = m'$ we have $Q(m')$.

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 \( m_1 \sim_{\text{low}} m_2 \):

1) \( \{c\}_{m_1} = \bot \) iff \( \{c\}_{m_2} = \bot \)

2) \( \{c\}_{m_1} = m_1' \) and \( \{c\}_{m_2} = m_2' \) implies \( m_1' \sim_{\text{low}} m_2' \)
Relational Hoare Logic - RHL

Precondition
Program$_1$ $\sim$ Program$_2$
Postcondition

$c_1 \sim c_2 : P \Rightarrow Q$

Precondition
(a logical formula)

Postcondition
(a logical formula)
Relational Assertions

\[ c_1 \sim c_2 : P \Rightarrow Q \]

Need to talk about variables of the two memories

\[ c_1 \sim c_2 : x⟨1⟩ \leq x⟨2⟩ \Rightarrow x⟨1⟩ \geq x⟨2⟩ \]

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) \( \{ c_1 \}_{m_1} = \bot \) iff \( \{ c_2 \}_{m_2} = \bot \)

2) \( \{ c_1 \}_{m_1} = m_1' \) and \( \{ c_2 \}_{m_2} = m_2' \) implies \( Q(m_1', m_2') \).

Is this easy to check?
Rules of Relational Hoare Logic

Skip

\[ \vdash \text{skip} \triangleright \text{skip} : P \Rightarrow P \]
Rules of Relational Hoare Logic

\[ \vdash \neg \text{abort} \rightarrow \text{true} \Rightarrow \text{false} \]
Rules of Relational Hoare Logic
Assignment

\[ \vdash_{x := e} x := e : P [e_1/x_1, e_2/x_2] \Rightarrow P \]