CS 591: Formal Methods in Security and Privacy
Probabilistic relational Hoare Logic

Marco Gaboardi
gaboardi@bu.edu

Alley Stoughton
stough@bu.edu
CS 591: Formal Methods in Security and Privacy
Probabilistic relational Hoare Logic

Marco Gaboardi
gaboardi@bu.edu

Alley Stoughton
stough@bu.edu
Projects

By the end of the week, everyone should know what to work on for the project.

If you don’t know yet what you want to work on, let’s schedule a time by email to zoom with Alley and me about projects ideas.
From the previous classes
Information Flow Control

We want to guarantee that confidential inputs do not flow to nonconfidential outputs.
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
Noninterference as a Relational Property

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

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

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

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

If we can derive $\vdash c_1 \sim c_2 : P \implies Q$ through the rules of the logic, then the quadruple $c_1 \sim c_2 : P \implies Q$ is valid.
Relative Completeness

If a quadruple $c_1 \sim c_2 \vdash P \Rightarrow Q$ 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 $\vdash c_1 \sim c_2 \vdash P \Rightarrow Q$ through the rules of the logic.
Soundness and completeness with respect to Hoare Logic

\[ \vdash_{RHL} c_1 \sim c_2 : P \Rightarrow Q \]

iff

\[ \vdash_{HL} c_1 ; c_2 : P \Rightarrow Q \]

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

A program \texttt{prog} is probabilistically noninterferent if and only if, whenever we run it on two low equivalent memories \texttt{m}_1 and \texttt{m}_2 we have that the probabilistic distributions we get as outputs are the same on public outputs.
c is probabilistically noninterferent if and only if for every $m_1 \sim_{\text{low}} m_2 : \{c\}_{m_1}=\mu_1$ and $\{c\}_{m_2}=\mu_2$ implies $\mu_1 \sim_{\text{low}} \mu_2$
An example

OneTimePad\[(m : \text{private } \text{msg}) : \text{public } \text{msg} \]
key := $ \text{Uniform} \{(0,1)^n\};$
\text{cipher} := \text{msg xor key};$
return \text{cipher}$

Learning a ciphertext does not change any a priori knowledge about the likelihood of messages.
Semantics of Commands

This is defined on the structure of commands:

\[
\begin{align*}
\{\text{abort}\}_m &= \emptyset & \{\text{skip}\}_m &= \text{unit}(m) \\
\{x:=e\}_m &= \text{unit}(m[x\leftarrow\{e\}_m]) \\
\{x:=d\}_m &= \text{let } a=\{d\}_m \text{ in } \text{unit}(m[x\leftarrow a]) \\
\{c;c'\}_m &= \text{let } m'=\{c\}_m \text{ in } \{c'\}_m' \\
\{\text{if } e \text{ then } c_t \text{ else } c_f\}_m &= \{c_t\}_m \text{ if } \{e\}_m=\text{true} \\
\{\text{if } e \text{ then } c_t \text{ else } c_f\}_m &= \{c_f\}_m \text{ if } \{e\}_m=\text{false} \\
\{\text{while } e \text{ do } c\}_m &= \sup_{n\in\text{Nat}} \mu_n \\
\mu_n &= \text{let } m'=\{(\text{while}^n e \text{ do } c)\}_m \text{ in } \{\text{if } e \text{ then } \text{abort}\}_m'
\end{align*}
\]
Revisiting the example

```plaintext
OneTimePad(m : private msg) : public msg
key := $ Uniform({0,1}^n);
cipher := msg xor key;
return cipher
```

Revisiting the example

<table>
<thead>
<tr>
<th>OneTimePad(m : private msg) : public msg</th>
</tr>
</thead>
<tbody>
<tr>
<td>key := $ Uniform({0,1}^n);</td>
</tr>
<tr>
<td>cipher := msg xor key;</td>
</tr>
<tr>
<td>return cipher</td>
</tr>
</tbody>
</table>

How can we prove that this is noninterferent?
Revisiting the example

\textbf{OneTimePad}(m : private msg) : public msg
key := Uniform({0,1}\textsuperscript{n});
cipher := msg xor key;
return cipher
Revisiting the example

```
OneTimePad(m : private msg) : public msg
key := Uniform({0,1}^n);
cipher := msg xor key;
return cipher
```

\[ m_1 \quad m_2 \]
Revisiting the example

\[ \text{OneTimePad}(m : \text{private msg}) : \text{public msg} \]

\[
\begin{align*}
\text{key} & :\text{Uniform}\left(\{0,1\}^n\right); \\
\text{cipher} & :\text{msg xor key}; \\
\text{return cipher}
\end{align*}
\]

\[ m_1 \oplus k \]

\[ m_1 \quad m_2 \]
Revisiting the example

OneTimePad(m : private msg) : public msg
key := Uniform({0,1}^n);
cipher := msg xor key;
return cipher

Suppose we can now chose the key for m₂. What could we choose?
Revisiting the example

\[
\text{OneTimePad}(m : \text{private msg}) : \text{public msg} \\
\text{key} := \$ \text{Uniform}\{0,1\}^n; \\
cipher := \text{msg xor key}; \\
return \text{cipher}
\]

\[
m_1 \oplus k \\
m_2 \oplus (m_1 \oplus k \oplus m_2)
\]

Suppose we can now chose the key for \(m_2\). What could we choose?
Properties of xor

\[ c \oplus (a \oplus c) = a \]
Properties of xor

\[ c \oplus (a \oplus c) = a \]

Example:

\[ 100 \oplus (101 \oplus 100) = \]
\[ 100 \oplus 001 = 101 \]
Revisiting the example

```plaintext
OneTimePad(m : private msg) : public msg
    key := Uniform({0,1}^n);
    cipher := msg xor key;
    return cipher
```

Applying the property above
Revisiting the example

OneTimePad(m : private msg) : public msg
key := Uniform({0,1}^n);
cipher := msg xor key;
return cipher

\( m_1 \) \hspace{1cm} \( m_2 \)

Applying the property above
Revisiting the example

OneTimePad($m : \text{private msg}) : \text{public msg}$
key := Uniform($\{0,1\}^n$);
cipher := msg xor key;
return cipher

Applying the property above

$m_1 \oplus k$

$m_1$

$m_2$

Applying the property above
Revisiting the example

<table>
<thead>
<tr>
<th>OneTimePad(m : private msg) : public msg</th>
</tr>
</thead>
<tbody>
<tr>
<td>key := Uniform({0,1}^n);</td>
</tr>
<tr>
<td>cipher := msg xor key;</td>
</tr>
<tr>
<td>return cipher</td>
</tr>
</tbody>
</table>

Applying the property above:

\[
m_1 \oplus k \\
\downarrow
\]

\[
m_1 \oplus k
\]

\[
m_2 \oplus k \\
\downarrow
\]

\[
m_1 \oplus k
\]

Applying the property above
Coupling

$\mu_1$

$\mu_2$
Coupling

$\mu_1$

$\mu_2$
Example of Our Coupling

\[ k = 10 \oplus k \oplus 00 \]
Example of Our Coupling

\[ k = 10 \oplus k \oplus 00 \]

<table>
<thead>
<tr>
<th></th>
<th>OO</th>
<th>O1</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td></td>
<td></td>
<td>0.25</td>
<td></td>
</tr>
<tr>
<td>O1</td>
<td></td>
<td></td>
<td></td>
<td>0.25</td>
</tr>
<tr>
<td>10</td>
<td>0.25</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td>0.25</td>
</tr>
</tbody>
</table>
Coupling formally

Given two distributions $\mu_1 \in \mathcal{D}(A)$, and $\mu_2 \in \mathcal{D}(B)$, a coupling between them is a joint distribution $\mu \in \mathcal{D}(A \times B)$ whose marginal distributions are $\mu_1$ and $\mu_2$, respectively.

$$
\pi_1(\mu)(a) = \sum_b \mu(a, b) \quad \pi_2(\mu)(b) = \sum_a \mu(a, b)
$$
Today:
Probabilistic Relational Hoare Logic
Probabilistic Relational Hoare Quadruples

Precondition
Program₁ ~ Program₂
Postcondition

\[ c₁ \sim c₂ : P \Rightarrow Q \]

Precondition (a logical formula)

Probabilistic Program
Probabilistic Program
Postcondition (a logical formula)
Validity of Probabilistic 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: $\{c_1\}_{m_1} = \mu_1$ and $\{c_2\}_{m_2} = \mu_2$ implies $Q(\mu_1, \mu_2)$. 
Validity of Probabilistic Hoare quadruple

We say that the quadruple $\mathbf{c}_1 \sim \mathbf{c}_2 : P \Rightarrow Q$ is valid if and only if for every pair of memories $m_1, m_2$ such that $P(m_1, m_2)$ we have:

$\{\mathbf{c}_1\}_{m_1} = \mu_1$ and $\{\mathbf{c}_2\}_{m_2} = \mu_2$ implies $Q(\mu_1, \mu_2)$.

Is this correct?!?
Relational Assertions

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

- Logical formula over pair of memories (i.e. relation over memories)
- Logical formula over ????
Given two distributions $\mu_1 \in D(A)$, and $\mu_2 \in D(B)$, an $R$-coupling between them, for $R \subseteq A \times B$, is a joint distribution $\mu \in D(A \times B)$ such that:

1) the marginal distributions of $\mu$ are $\mu_1$ and $\mu_2$, respectively,

2) the support of $\mu$ is contained in $R$. That is, if $\mu(a, b) > 0$, then $(a, b) \in R$. 

$R$-Coupling
Relational lifting of a predicate

We say that two subdistributions $\mu_1 \subseteq D(A)$ and $\mu_2 \subseteq D(B)$ are in the relational lifting of the relation $R \subseteq A \times B$, denoted $\mu_1 R^* \mu_2$ if and only if there exist a subdistribution $\mu \subseteq D(A \times B)$ such that:

1) if $\mu(a, b) > 0$, then $(a, b) \in Q$.

2) $\pi_1(\mu) = \mu_1$ and $\pi_2(\mu) = \mu_2$
Relational lifting of a predicate

We say that two subdistributions $\mu_1 \subseteq D(A)$ and $\mu_2 \subseteq D(B)$ are in the relational lifting of the relation $R \subseteq A \times B$, denoted $\mu_1 R^* \mu_2$ if and only if there exist a subdistribution $\mu \subseteq D(A \times B)$ such that:

1) if $\mu(a, b) > 0$, then $(a, b) \in Q$.
2) $\pi_1(\mu) = \mu_1$ and $\pi_2(\mu) = \mu_2$

Does it remind you something?
Validity of Probabilistic 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:

$\{c_1\}_{m_1} = \mu_1$ and $\{c_2\}_{m_2} = \mu_2$ implies $Q^*(\mu_1, \mu_2)$. 
Probabilistic Relational Hoare Logic

Skip

\[ \vdash \neg \text{skip} \sim \text{skip} : P \Rightarrow P \]
\[\vdash x_1 := e_1 \sim x_2 := e_2 : \]
\[P[e_1<1>/x_1<1>, e_2<2>/x_2<2>] \Rightarrow P\]
\[ \vdash c_1 \sim c_2 : P \Rightarrow R \quad \vdash c_1' \sim c_2' : R \Rightarrow S \]

\[ \dashedline \]

\[ \vdash c_1 ; c_1' \sim c_2 ; c_2' : P \Rightarrow S \]
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 $R$. 
\[
\frac{P \Rightarrow (e_1<1> \iff e_2<2>)
\quad \vdash c_1 \sim c_2 : e_1<1> \land P \Rightarrow Q
\quad \vdash c_1' \sim c_2' : \neg e_1<1> \land P \Rightarrow Q}{\vdash : P \Rightarrow Q}
\]

if \(e_1\) then \(c_1\) else \(c_1'\)

if \(e_2\) then \(c_2\) else \(c_2'\)
Probabilistic Relational Hoare Logic

While

\[ P \Rightarrow (e_1<1> \iff e_2<2>) \]

\[ \vdash c_1 \sim c_2 : e_1<1> \land P \Rightarrow P \]

\[ \vdash \quad ~ \quad : P \Rightarrow P \land \neg e_1<1> \]

while \( e_1 \) do \( c_1 \)

while \( e_2 \) do \( c_2 \)
Probabilistic Relational Hoare Logic
If-then-else - left

\[ \vdash c_1 \sim c_2 : e<1> \land P \Rightarrow Q \]

\[ \vdash c_1' \sim c_2 : \neg e<1> \land P \Rightarrow Q \]

\[ \vdash \text{if } e \text{ then } c_1 \text{ else } c_1' \]
\[ \vdash \sim c_2 \quad : P \Rightarrow Q \]
Probabilistic Relational Hoare Logic
If-then-else - right

\[ \vdash \neg c_1 \sim c_2 : e < 2 > \land P \Rightarrow Q \]

\[ \vdash \neg c_1 \sim c_2' : \neg e < 2 > \land P \Rightarrow Q \]

\[ \vdash \sim c_1 \quad : P \Rightarrow Q \]

\[ \text{if } e \text{ then } c_2 \text{ else } c_2' \]
\[ \vdash x := e \sim \text{skip} : \]
\[ P[e<1>/x<1>] \implies P \]
How about the random assignment?
Probabilistic Relational Hoare Logic

Random Assignment

\[ \vdash x_1 := \$d_1 \sim x_2 := \$d_2 : ?? \]
We would like to have:

\[ P(m_1,m_2) \]
\[ \Rightarrow \]

\[ \text{let } a = \{d_1\}_{m_1} \text{ in unit}(m_1[x_1 \leftarrow a]) \]
\[ Q^* \]
\[ \text{let } a = \{d_2\}_{m_2} \text{ in unit}(m_2[x_2 \leftarrow a]) \]

\[ \vdash x_1 := \_ \quad d_1 \sim x_2 := \_ \quad d_2 : \ P \Rightarrow Q \]
We would like to have:

\[ P(m_1, m_2) \]

\[ \Rightarrow \]

let \( a = \{d_1\}_{m_1} \) in unit\((m_1[x_1 \leftarrow a])\)

Q*

let \( a = \{d_2\}_{m_2} \) in unit\((m_2[x_2 \leftarrow a])\)

\[ \vdash x_1 \ := \$ \ d_1 \sim \ x_2 \ := \$ \ d_2 : \ P \ \Rightarrow \ Q \]

What is the problem with this rule?