CS 591: Formal Methods in Security and Privacy

Approximate probabilistic relational Hoare Logic

Marco Gaboardi
gaboardi@bu.edu

Alley Stoughton
stough@bu.edu

Q&A

To increase interactivity, I will ask more question to each one of you.

It is not a test, you can always answer “pass!”

Assignments

Remember that the third assignment was due yesterday. If you are still working on it, no problem, but please do let us know.

Recording

This is a reminder that we will record the class and we will post the link on Piazza.

This is also a reminder to myself to start recording!

From the previous classes

An example

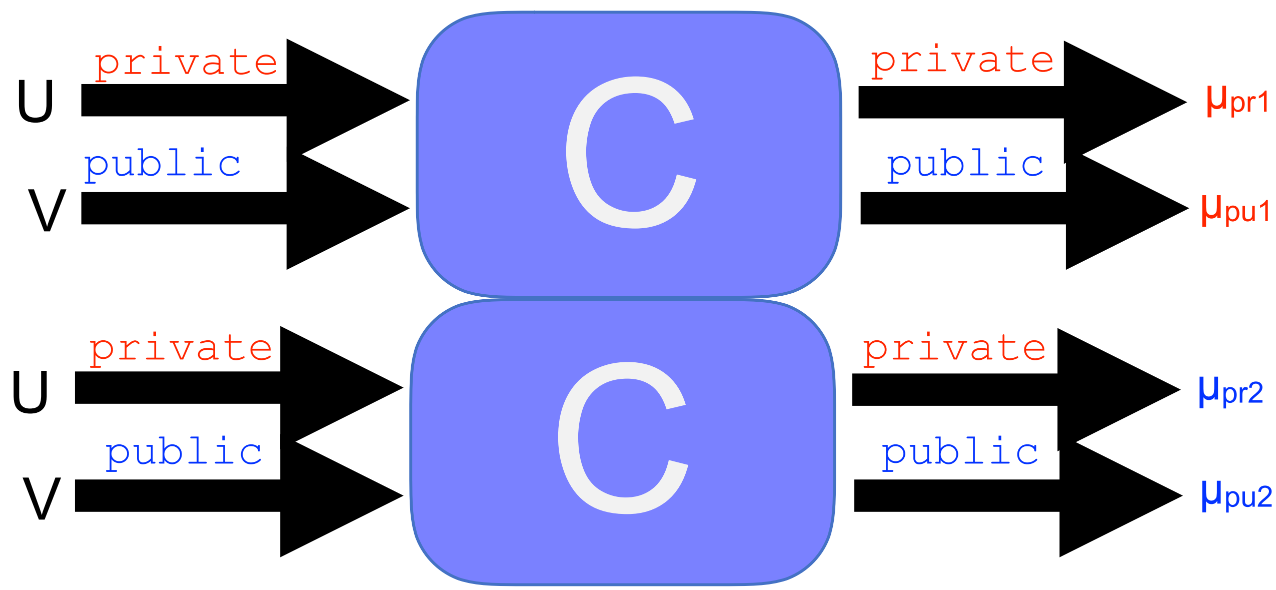
```
OneTimePad(m : private msg) : public msg
   key :=$ Uniform({0,1}^n);
   cipher := msg xor key;
   return cipher
```

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

Probabilistic Noninterference as a Relational Property

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

$\{c\}_{m1} = \mu_1$ and $\{c\}_{m2} = \mu_2$ implies $\mu_1 \sim_{low} \mu_2$



Probabilistic Relational Hoare Quadruples

Precondition

$\mathtt{Program_1} \sim \mathtt{Program_2}$

Postcondition

Precondition (a logical formula)

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

Probabilistic Program

Probabilistic Program

Postcondition (a logical formula)

R-Coupling

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 μ are μ_1 and μ_2, respectively,
2) the support of μ is contained in R. That is, if $\mu(a,b)>0$, then $(a,b) \in R$.

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:

$\{c_1\}_{m1} = \mu_1$ and $\{c_2\}_{m2} = \mu_2$ implies $Q^*(\mu_1, \mu_2)$.

Consequences of Coupling

Given the following pRHL judgment

$$\vdash c_1 \sim c_2 : \text{True} \Rightarrow Q$$

We have that:

if $Q \Rightarrow (R\langle 1\rangle \iff S\langle 2\rangle)$, then $\Pr[c_1 : R] = \Pr[c_2 : S]$

if $Q \Rightarrow (R\langle 1\rangle \Rightarrow S\langle 2\rangle)$, then $\Pr[c_1 : R] \leq \Pr[c_2 : S]$

A more realistic example

```
StreamCipher(m : private msg[n]) : public msg[n]
   pkey :=$ PRG(Uniform({0,1}^k));
   cipher := msg xor pkey;
   return cipher
```

Properties of PRG

We would like the PRG to increase the number of random bits but also to guarantee the result to be (almost) random.

We can express this as:

$\boxed{\text{PRG}: \{0,1\}^k \rightarrow \{0,1\}^n \text{ for } n>k}$

$\boxed{\text{PRG}(\text{Uniform}(\{0,1\}^k) \approx \text{Uniform}(\{0,1\}^n}$

How can we measure the similarity between the result of PRG and the uniform distribution?

Statistical distance

We say that two distributions $\mu_1, \mu_2 \in D(A)$, are at statistical distance δ if and only if:

$$\Delta(\mu 1,\mu 2)=\max_{E\subseteq A} \mid \mu_1(E)-\mu_2(E) \mid = \delta$$

For discrete distributions the statistical distance can also be characterized as:

$$\Delta(\mu 1,\mu 2)= 1/2 \sum_{a\in A} \mid \mu_1(a)-\mu_2(a) \mid$$

Properties of PRG

We would like the PRG to increase the number of random bits but also to guarantee the result to be (almost) random.

We can express this as:

$$\boxed{PRG: \{0,1\}^k \rightarrow \{0,1\}^n \text{ for } n>k}$$

$$\boxed{\Delta(PRG(Uniform(\{0,1\}^k),Uniform(\{0,1\}^n) \leq 2^{-n}}$$

In fact this is a too strong requirement - usually we require that every polynomial time adversary cannot distinguish the two distributions in statistical distance

How can we prove this secure?

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

~

```
StreamCipher(m : private msg[n])
            : public msg[n]
   pkey :=$ PRG(Uniform({0,1}^k));
   cipher := msg xor pkey;
   return cipher
```

How can we prove this secure?

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

~

```
StreamCipher(m : private msg[n])
             : public msg[n]
   pkey :=$ PRG(Uniform({0,1}^k));
   cipher := msg xor pkey;
   return cipher
```

m

m

How can we prove this secure?

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

~

```
StreamCipher(m : private msg[n])
             : public msg[n]
   pkey :=$ PRG(Uniform({0,1}^k));
   cipher := msg xor pkey;
   return cipher
```

$$m \;\Big\downarrow\; m \oplus k$$

$$m$$

How can we prove this secure?

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

~

```
StreamCipher(m : private msg[n])
            : public msg[n]
   pkey :=$ PRG(Uniform({0,1}^k));
   cipher := msg xor pkey;
   return cipher
```

$$\begin{tikzcd} m \arrow[d] \\ m \oplus k \end{tikzcd} \qquad \begin{tikzcd} m \arrow[d] \\ m \oplus pk \end{tikzcd}$$

How can we prove this secure?

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

~

```
StreamCipher(m : private msg[n])
             : public msg[n]
   pkey :=$ PRG(Uniform({0,1}^k));
   cipher := msg xor pkey;
   return cipher
```

$m \qquad\qquad m$

$$\Delta(\, m\oplus k \;,\; m\oplus pk \,) \le \delta$$

Today:
approximate probabilistic noninterference

How to reason formally about this formally?

Approximate Probabilistic Relational Hoare Logic

Indistinguishability parameter

Precondition (a logical formula)

$$\vdash_{\delta} c_1 \sim c_2 : P \Rightarrow Q$$

Probabilistic Program

Probabilistic Program

Postcondition (a logical formula)

How can we define validity?

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:

$\{c_1\}_{m1} = \mu_1$ and $\{c_2\}_{m2} = \mu_2$ implies $Q^*(\mu_1, \mu_2)$.

R–δ–Coupling

Given two distributions $\mu_1 \in D(A)$, and $\mu_2 \in D(B)$, we have an R-δ-coupling between them, for $R \subseteq A \times B$ and $0 \leq \delta \leq 1$, if there are two joint distributions $\mu_L, \mu_R \in D(A \times B)$ such that:

1) $\pi_1(\mu_L)=\mu_1$ and $\pi_2(\mu_R)=\mu_2$,
2) the support of μ_L and μ_R is contained in R. That is, if $\mu_L(a,b)>0$, then $(a,b) \in R$, and if $\mu_R(a,b)>0$, then $(a,b) \in R$.
3) $\Delta(\mu_L, \mu_R) \leq \delta$

Approximate 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 δ–lifting of the relation $R \subseteq A x B$, denoted $\mu_1 \; R_\delta{*} \; \mu_2$ if and only if there exist an R-coupling between them.

Validity of approximate Probabilistic Hoare judgments

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

$\{\mathtt{c}_1\}_{\mathtt{m1}} = \mu_1$ and $\{\mathtt{c}_2\}_{\mathtt{m2}} = \mu_2$ implies

$\mathtt{Q}_\delta{}^*(\mu_1, \mu_2)$.

Example of R-δ-Coupling

μ_1

00	0.25
01	0.25
10	0.25
11	0.25

μ_2

00	0.20
01	0.25
10	0.25
11	0.30

Example of R-δ-Coupling

μ_1

00	0.25
01	0.25
10	0.25
11	0.25

$$R(a,b) = \{a=b\}$$

μ_2

00	0.20
01	0.25
10	0.25
11	0.30

Example of R-δ-Coupling

μ_1

00	0.25
01	0.25
10	0.25
11	0.25

$$R(a,b) = \{a=b\}$$

μ_2

00	0.20
01	0.25
10	0.25
11	0.30

μ_L	00	01	10	11
00	0.25			
01		0.25		
10			0.25	
11				0.25

μ_R	00	01	10	11
00	0.20			
01		0.25		
10			0.25	
11				0.30

Example of R-δ-Coupling

μ_1

00	0.25
01	0.25
10	0.25
11	0.25

μ_2

00	0.20
01	0.25
10	0.25
11	0.30

$$R(a,b) = \{a=b\}$$

μ_L	00	01	10	11
00	0.25			
01		0.25		
10			0.25	
11				0.25

μ_R	00	01	10	11
00	0.20			
01		0.25		
10			0.25	
11				0.30

$$\Delta(\mu_L, \mu_R) = 0.05$$

Example of R-δ-Coupling

μ_1

OO	0.2
O1	0.25
1O	0.25
11	0.3

μ_2

OO	0
O1	0.40
1O	0
11	0.6

Example of R-δ-Coupling

μ_1

00	0.2
01	0.25
10	0.25
11	0.3

$$R(a,b) = \{a \le b\}$$

μ_2

00	0
01	0.40
10	0
11	0.6

Example of R-δ-Coupling

μ_1

OO	0.2
O1	0.25
1O	0.25
11	0.3

$$R(a,b) = \{a \leq b\}$$

μ_2

OO	0
O1	0.40
1O	0
11	0.6

μ_L	OO	O1	1O	11
OO		0.20		
O1		0.25		
1O				0.25
11				0.30

μ_R	OO	O1	1O	11
OO		0.20		
O1		0.20		
1O				0.3
11				0.3

Example of R-δ-Coupling

μ_1

00	0.2
01	0.25
10	0.25
11	0.3

$$R(a,b) = \{a \le b\}$$

μ_2

00	0
01	0.40
10	0
11	0.6

μ_L	00	01	10	11
00		0.20		
01		0.25		
10				0.25
11				0.30

μ_R	00	01	10	11
00		0.20		
01		0.20		
10				0.3
11				0.3

$$\Delta(\mu_L, \mu_R) = 0.05$$

A more realistic example

```
StreamCipher(m : private msg[n]) : public msg[n]
   pkey :=$ PRG(Uniform({0,1}^k));
   cipher := msg xor pkey;
   return cipher
```

Example of R-δ-Coupling

μ_1

OO	0.25
O1	0.25
1O	0.25
11	0.25

μ_2

OO	0
O1	0
1O	0.5
11	0.5

Example of R-δ-Coupling

μ_1

OO	0.25
O1	0.25
1O	0.25
11	0.25

$$R(a,b) = \{a=b\}$$

μ_2

OO	0
O1	0
1O	0.5
11	0.5

Example of R-δ-Coupling

μ_1

00	0.25
01	0.25
10	0.25
11	0.25

$$R(a,b) = \{a=b\}$$

μ_2

00	0
01	0
10	0.5
11	0.5

μ_L	00	01	10	11
00	0.25			
01		0.25		
10			0.25	
11				0.25

μ_R	00	01	10	11
00	0			
01		0		
10			0.5	
11				0.5

Example of R-δ-Coupling

μ_1

00	0.25
01	0.25
10	0.25
11	0.25

$$R(a,b) = \{a=b\}$$

μ_2

00	0
01	0
10	0.5
11	0.5

μ_L	00	01	10	11
00	0.25			
01		0.25		
10			0.25	
11				0.25

μ_R	00	01	10	11
00	0			
01		0		
10			0.5	
11				0.5

$$\Delta(\mu_L, \mu_R) = 0.5$$

Probabilistic Relational Hoare Logic
Skip

$$\frac{}{\vdash_0 \texttt{skip} \sim \texttt{skip} : P \Rightarrow P}$$

Probabilistic Relational Hoare Logic

Assignment

$$\vdash_{0} x_1 := e_1 \sim x_2 := e_2 : P[e_1\langle 1\rangle / x_1\langle 1\rangle, e_2\langle 2\rangle / x_2\langle 2\rangle] \Rightarrow P$$

Probabilistic Relational Hoare Logic

Composition

$$\dfrac{\vdash_0 c_1 \sim c_2 : P \Rightarrow R \qquad \vdash_0 c_1' \sim c_2' : R \Rightarrow S}{\vdash_0 c_1; c_1' \sim c_2; c_2' : P \Rightarrow S}$$

Probabilistic Relational Hoare Logic
Composition

$$\frac{\vdash_{\delta_1} c_1 \sim c_2 : P \Rightarrow R \qquad \vdash_{\delta_2} c_1' \sim c_2' : R \Rightarrow S}{\vdash_{\delta_1+\delta_2} c_1;c_1' \sim c_2;c_2' : P \Rightarrow S}$$

Probabilistic Relational Hoare Logic

Consequence

$$\frac{P \Rightarrow S \qquad \vdash_{\delta 1} c_1 \sim c_2 : S \Rightarrow R \qquad R \Rightarrow Q \qquad \delta 1 \leq \delta 2}{\vdash_{\delta 2} c_1 \sim c_2 : P \Rightarrow Q}$$

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.

We can relax $\delta 1$, i.e. replace it by something larger, in this case $\delta 2$.

Probabilistic Relational Hoare Logic

If-then-else

$$P \Rightarrow (e_1\langle 1\rangle \Leftrightarrow e_2\langle 2\rangle)$$

$$\vdash_\delta c_1 \sim c_2 : e_1\langle 1\rangle \wedge P \Rightarrow Q$$

$$\vdash_\delta c_1' \sim c_2' : \neg e_1\langle 1\rangle \wedge P \Rightarrow Q$$

$$\vdash_\delta \begin{array}{c} \texttt{if } e_1 \texttt{ then } c_1 \texttt{ else } c_1' \\ \sim \\ \texttt{if } e_2 \texttt{ then } c_2 \texttt{ else } c_2' \end{array} : P \Rightarrow Q$$

Probabilistic Relational Hoare Logic

If-then-else - left

$$\vdash_{\delta} c_1 \sim c_2 : e\langle 1\rangle \wedge P \Rightarrow Q$$

$$\vdash_{\delta} c_1' \sim c_2 : \neg e\langle 1\rangle \wedge P \Rightarrow Q$$

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

Probabilistic Relational Hoare Logic

A specific rule for PRG

$$\vdash_{2\char`^-n}\ \mathtt{x}_1 \ \mathtt{:=\$}\ \mathtt{Uniform}(\{0,1\}^n) \sim$$

$$\mathtt{x}_2\ \mathtt{:=\$}\ \mathtt{PRG}(\mathtt{Uniform}(\{0,1\}^k))$$

$$:\ \mathtt{True} \Rightarrow \mathtt{x}_1\langle 1\rangle = \mathtt{x}_2\langle 2\rangle$$

How can we prove this secure?

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

~

```
StreamCipher(m : private msg[n])
             : public msg[n]
   pkey :=$ PRG(Uniform({0,1}^k));
   cipher := msg xor pkey;
   return cipher
```

We can apply the PRG rule, the composition rule, and the assignment rule and prove:

$$\vdash_{2^{-n}} \mathtt{OneTimePad} \sim \mathtt{StreamCipher} : m\langle 1\rangle = m\langle 2\rangle \Rightarrow c\langle 1\rangle = c\langle 2\rangle$$