

Measuring information flow due to branching behaviour in programs

P. Malacaria (Queen Mary, University of London)

S. Hunt (City University, London)

D. Clark (King's College London)

Overview

- ❑ Brief primer on information theory
- ❑ sketch of how our analysis works
- ❑ derivation of inference rules for analysing equality tests
- ❑ small example
- ❑ derivation of inference rules for analysing if statements
- ❑ example and discussion
- ❑ conclusions

The Problem

- ❑ **problem**: Static analysis of quantification of interference between two variables caused by running a while program.
- ❑ **application**: Leakage of secrets from confidential to non-confidential variables in a program.
- ❑ **attack**: Limited. Attacker knows non-confidential inputs/outputs and program text. Timing not considered.

Our Solution

- ❑ **calculation**: Use information theory to find average number of “bits of information” leaked across all runs of the program.
- ❑ **requires**: statistical information – probability distribution on inputs (or some information about this).
- ❑ **focus**: Languages with a ‘simple function semantics’: program viewed as a transformation of inputs to outputs.
- ❑ **simplification**: Programs terminate.

Entropy and Information

- ❑ Surprisal of an event: $\log_2 \frac{1}{p}$.
- ❑ Total information carried by a set of n events: weighted sum of surprisal values.

$$\mathcal{H} = \sum_{i=1}^n p_i \log \frac{1}{p_i}$$

Called *self information* or *entropy* of the set of events.

- ❑ Greatest when probability distribution is uniform, 0 when single value is certain.

Random variables and program points

- ❑ A random variable (or discrete random element in this case) is a total function $X : D \rightarrow R$. D and R are finite sets, D has a probability distribution.
- ❑ joint random variable: (X, Y) defined as $\langle X, Y \rangle$
- ❑ Entropy of a random variable X :

$$\mathcal{H}(X) = \sum_{x \in R} p(x) \log \frac{1}{p(x)}$$

- ❑ Associate random variables with expressions, particularly program variables, at program points within a program.

Conditional Entropy

- $P((X \upharpoonright (Y = y)) = x) = P(X = x|Y = y)$, where

$$P(X = x|Y = y) = \frac{p(x, y)}{p(y)}$$



$$\mathcal{H}(X|Y) = \sum_y p(y)\mathcal{H}(X \upharpoonright (Y = y))$$

- A key property of conditional information is that $\mathcal{H}(X|Y) \leq \mathcal{H}(X)$, with equality iff X and Y are independent.

Leakage

The amount of leakage of confidential information into variable X due to execution of the program:

$$\mathcal{L}(X) = \mathcal{H}(X^\omega | L^\iota)$$

Alternatively: information shared between final value of X and the initial value of H , given that the initial value of L was already known:

$$\mathcal{L}'(X) = \mathcal{I}(H^\iota; X^\omega | L^\iota)$$

See [Gray 91: Toward a mathematical foundation for information flow security]

Analysis for While Language

- Assume for vector of confidential variables, H at ι we know bounds on the entropy for H^ι . Default is $0 \leq \mathcal{H}(H) \leq N$ where N is number of bits in vector.
- For each non-confidential variable X calculate upper and lower bounds on $\mathcal{L}(X) = \mathcal{H}(X^\omega | L^\iota)$.

Analyse for worst case choice of L^ι

- ❑ likely little or no information available about low inputs
- ❑ low inputs may be in *control* of attacker
- ❑ For random variable X , let $X_\lambda = (X|L = \lambda)$
- ❑ Analysis calculates bounds on $\mathcal{H}(X_\lambda^\omega)$ given bounds on $\mathcal{H}(H_\lambda^\iota)$
- ❑ require bounds which hold for all λ
- ❑ Proposition:

$$(\forall \lambda . a \leq \mathcal{H}(X_\lambda^\omega) \leq b \Rightarrow (a \leq \mathcal{H}(X^\omega|L^\iota) \leq b))$$

(average is bounded by its smallest and largest terms)

- ❑ Assuming k -bit variables, the bounds $0 \leq \mathcal{H}(X^\iota | L^\iota = \lambda) \leq k$ are always valid
- ❑ For all low-security variables $X \in L$, $\mathcal{H}(X^\iota | L^\iota = \lambda) = 0$
- ❑ In most cases, it will be reasonable to assume that the initial values of H and L are independent, in which case $\mathcal{H}(X^\iota | L^\iota = \lambda) = \mathcal{H}(X^\iota)$ for all high-security variables $X \in H$, so the problem of calculating useful initial assumptions reduces to the problem of finding good estimates of the entropies of the high security inputs

Analyse tests of form $E_1 == E_2$

- ❑ find bounds on $\mathcal{H}(X == Y)$
- ❑ Observation: When p.d. of values for E_1 is close to uniform and p.d. for E_2 has only a few non-zero values, then *most of the time* $E_1 \neq E_2$
- ❑ make use of this observation:
- ❑ Suppose X a k -bit r.v. and $P(X = x) = q$.
- ❑ What is the maximum possible value for $\mathcal{H}(X)$? *This is when the most information will be leaked by evaluating the expression.*
- ❑ Call this $\mathcal{U}_k(q)$: the upper entropy for q in k bits.

Upper entropy for q in k -bits

- Assume $P(X = x) = q$
- Entropy at maximum for uniform distributions
- Max value for $\mathcal{H}(X)$ when $P(X = x')$ is a uniform p.d. for all $x' \neq x$.
- In a k -bit r.v. there are $2^k - 1$ such x' so applying definition of entropy we have
- $\mathcal{U}_k(q) \stackrel{\text{def}}{=} q \log \frac{1}{q} + (1 - q) \log \frac{2^k - 1}{1 - q}$
- We show that if $P(X = Y) = q$ then $\mathcal{U}_k(q)$ is an upper bound for the difference between $\mathcal{H}(X)$ and $\mathcal{H}(Y)$.

Proposition 1: $\mathcal{H}(X|Y) \leq \mathcal{U}_k(q)$

□

$$\begin{aligned} \mathcal{U}_k(q) &= q \log \frac{1}{q} + (1 - q) \log \frac{2^k - 1}{1 - q} = \\ &= q \log \frac{1}{q} + (1 - q) \log \left(\frac{1}{1 - q} \right) + (1 - q) \log(2^k - 1) = \\ &= \mathcal{B}(q) + (1 - q) \log(2^k - 1) \end{aligned}$$

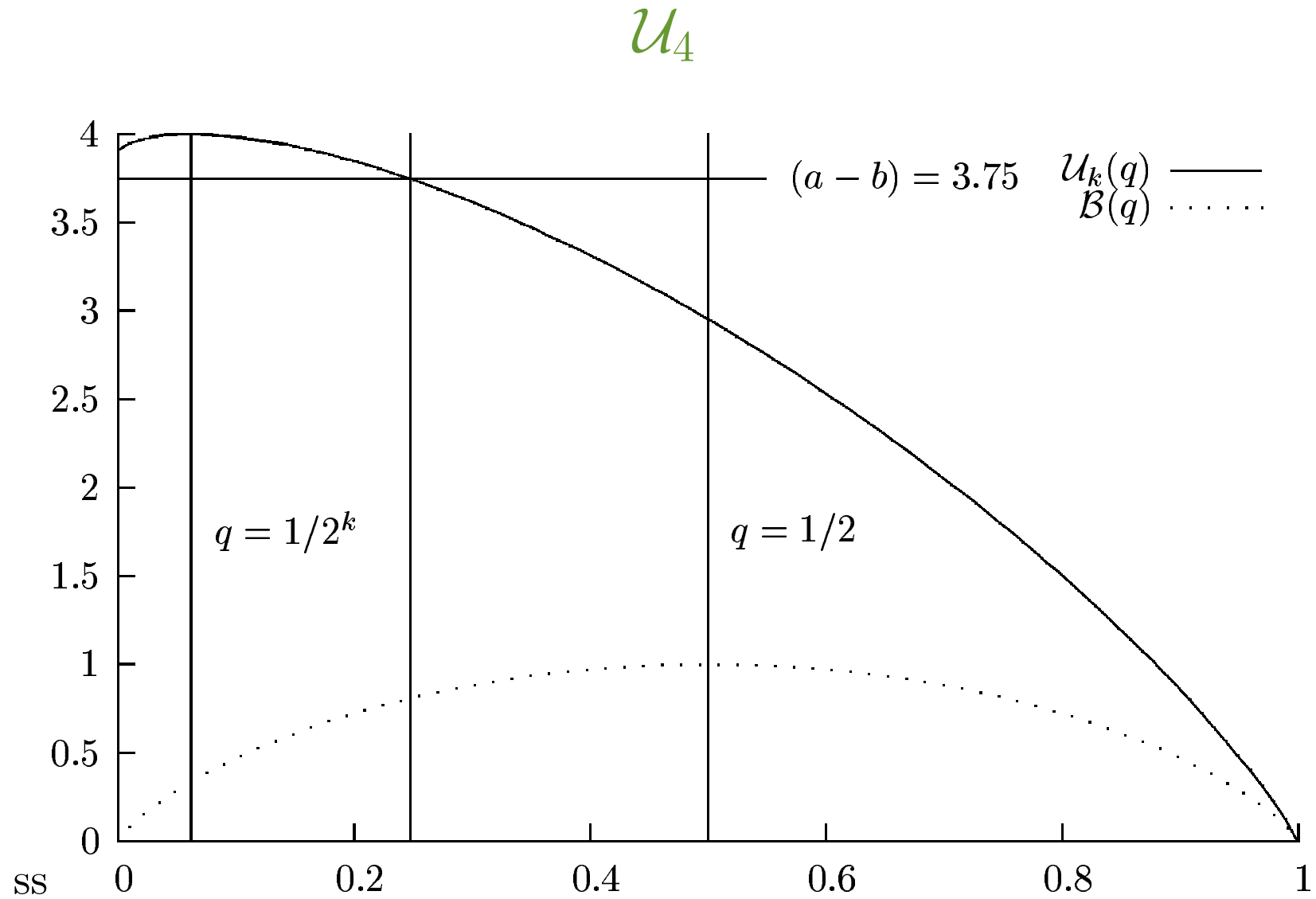
where $\mathcal{B}(q) \stackrel{\text{def}}{=} q \log \frac{1}{q} + (1 - q) \log \frac{1}{1 - q}$

□ Fano's inequality is well known in information theory:

□ $\mathcal{H}(X|Y) \leq \mathcal{B}(q) + (1 - q) \log(2^k - 1)$

Corollaries

- Since $\mathcal{H}(X|Y) = \mathcal{H}(X, Y) - \mathcal{H}(Y) \geq \mathcal{H}(X) - \mathcal{H}(Y)$, we have
- *Corollary 1* $\mathcal{H}(X) - \mathcal{H}(Y) \leq \mathcal{U}_k(q)$, and so
- *Corollary 2* If $a \leq \mathcal{H}(X)$ and $\mathcal{H}(Y) \leq b$ then $a - b \leq \mathcal{U}_k(q)$
- Recall we trying to establish bounds on $\mathcal{H}(X == Y)$ i.e. establish $\mathcal{B}(q)$ for $q = P(X = Y)$ and we don't know q



Proposition 2

- in interval $[1/2^k, 0.5]$, $\mathcal{U}_k(q)$ is decreasing and $\mathcal{B}(q)$ is increasing.
So
- $a - b = \mathcal{U}_k(q_0)$ iff $\mathcal{B}(q_0) = \text{Max}\{\mathcal{B}(q) | 1/2^k \leq q \leq 0.5, a - b \leq \mathcal{U}_k(q)\}$
- Want to show **If $a \leq \mathcal{H}(X)$ and $\mathcal{H}(Y) \leq b$ then**
 $\mathcal{H}(X == Y) \leq \mathcal{B}(q_0)$ for $1/2^k \leq q_0 \leq 0.5$ s.t. $a - b = \mathcal{U}_k(q_0)$
- But $\mathcal{H}(X == Y) = \mathcal{B}(q)$ means $q = P(X = Y)$ and so
proposition 2 follows from proposition 1 and its corollaries.

Safe Upper bounds for the upper bound

- ❑ To find best upper bound for $\mathcal{H}(X == Y)$ knowing a and b we need to solve equations of the form $\mathcal{U}_k(q) - (a - b) = 0$
- ❑ Use numerical methods
- ❑ Easier to work with safe approximations to this bound.
- ❑ Recall \mathcal{U}_k is a decreasing function and \mathcal{B} is an increasing function in the given interval
- ❑ Smaller values than $\mathcal{U}_k(q_0)$ give larger values for \mathcal{B} .
- ❑ **Corollary 3** If $a \leq \mathcal{H}(X)$ and $\mathcal{H}(Y) \leq b$ then $\mathcal{H}(X == Y) \leq \mathcal{B}(q)$ for all q $1/2^k \leq q \leq 0.5$ s.t. $a - b \geq \mathcal{U}_k(q)$

Some Inference rules

$$\frac{\Gamma \vdash E_1 : [a, -] \quad \Gamma \vdash E_2 : [-, b]}{\Gamma \vdash (E_1 == E_2) : [0, \mathcal{B}(q)]} \quad \frac{1}{2^k} \leq q \leq \frac{1}{2}, \mathcal{U}_k(q) \leq (a - b)$$

$$\frac{\Gamma \vdash E_1 : [-, b_1] \quad \Gamma \vdash E_2 : [-, b_2]}{\Gamma \vdash (E_1 == E_2) : [0, \min(1, b_1 + b_2)]}$$

$$\frac{\Gamma \vdash E : [a, b]}{\vdash \Gamma \{x := E\} x : [a, b]}$$

$$\frac{\vdash \Gamma \{C_1\} \Gamma' \quad \vdash \Gamma' \{C_2\} \Gamma''}{\vdash \Gamma \{C_1; C_2\} \Gamma''}$$

An example

- Consider the program P :

$Y = H; \text{ if } (Y == 0) \text{ then } X = 0 \text{ else } X = 1 \text{ fi}; Z = X$

with H high-security.

- Suppose that $k = 32$ and the input distribution makes H uniform over its 2^{32} values. Then rules for assignment and rule for equality tests give

- $H, Y: [32, 32] \vdash (Y == 0) : [0, \epsilon]$

- $\epsilon = \mathcal{B}(1/2^{32}) \approx 7.8 \times 10^{-9}$

Bounding leakage into a variable via execution of an if statement

- $C = \text{if } (e == \text{tt}) \text{ then } C_1 \text{ else } C_2$
- Assume we have:
 - ➔ bounds on the leakage due to evaluation of the control expression and
 - ➔ bounds on the leakage into the variable as a result of the execution of the individual branches.
 - ➔ the control expression is an equality test
 - ➔ the test is not true very often (i.e. the probability $p(e = \text{false}) \approx 1$)

Upper bounds

- $\mathcal{H}(F) \leq \mathcal{H}(F, e) = \mathcal{H}(F|e) + \mathcal{H}(e)$ (standard identity)
- F is r.v. associated with leakage into a variable as a result of executing C
- e is r.v. associated with evaluation of control expression for the if statement
- assume can calculate bound for $\mathcal{H}(e)$, want to bound $\mathcal{H}(F|e)$
- $p(e = \text{false}) \stackrel{\text{def}}{=} r$
- can expand $\mathcal{H}(F|e)$:

$$\begin{aligned} \mathcal{H}(F|e) &= r\mathcal{H}(F|e = \text{ff}) + (1 - r)\mathcal{H}(F|e = \text{tt}) \\ &= r\mathcal{H}(C_2|e = \text{ff}) + (1 - r)\mathcal{H}(C_1|e = \text{tt}) \end{aligned}$$

Upper bounds

- Assume entropy of true branch is bounded by k (maximum possible for F). Since r close to 1, this branch makes small weighted contribution
- Can show weighted other branch bounded by $\mathcal{H}(C_2)$

$$\begin{aligned}
 r\mathcal{H}(C_2|e = ff) &\leq r\mathcal{H}(C_2|e = ff) + (1 - r)\mathcal{H}(C_2|e = tt) \\
 &= \mathcal{H}(C_2|e) \\
 &\leq \mathcal{H}(C_2)
 \end{aligned}$$

- So $\mathcal{H}(F) \leq \mathcal{H}(e) + \mathcal{H}(C_2) + (1 - r)k$

Lower bounds

- ❑ Lower bounds sometimes important: e.g. equality tests
- ❑ To get a good lower bound need to use \mathcal{L}_1 inequality (well known in information theory)
- ❑ Define the \mathcal{L}_1 distance between two probability distributions, p and q on the same event space, written $\|p - q\|_1$, as follows

$$\|p - q\|_1 \stackrel{\text{def}}{=} \sum_x |p(x) - q(x)|$$

Lower bounds

□ \mathcal{L}_1 Inequality:

$$\|p - q\|_1 \leq \frac{1}{2} \Rightarrow |\mathcal{H}(p) - \mathcal{H}(q)| \leq \|p - q\|_1 \log \left(\frac{|X|}{\|p - q\|_1} \right)$$

where $|X|$ is the size of the event space for both p and q .

□ Manipulate the RHS of the inequality in the consequent as follows:

$$\begin{aligned} \|p - q\|_1 \log \left(\frac{|X|}{\|p - q\|_1} \right) &= \|p - q\|_1 (\log |X| - \log \|p - q\|_1) \\ &= \|p - q\|_1 (k - \log \|p - q\|_1) \end{aligned}$$

Lower bounds

- ❑ How do we use this?
- ❑ For a generic program variable z (interpreted as a random variable) we consider
 - ➔ $p(x)$ is the probability of z assuming a particular value x by running \mathbf{C}_2
 - ➔ $q(x)$ is the probability of z assuming a particular value x by running \mathbf{C}_2 given that $e == ff$, i.e. $q(x) = p(x|e = ff)$.
- ❑ We then show $\|p - q\|_1 \leq \frac{1}{r} - r$ (but not in this talk).
- ❑ Since we assume r is close to 1 then $\frac{1}{r} - r \leq \frac{1}{2}$, satisfying the LHS of \mathcal{L}_1 inequality.

Lower bounds

- Applying the \mathcal{L}_1 inequality:

$$\begin{aligned}
 |\mathcal{H}(p) - \mathcal{H}(q)| &\leq \|p - q\|_1 (k - \log \|p - q\|_1) \\
 &\leq \left(\frac{1}{r} - r\right) (k - \log \left(\frac{1}{r} - r\right)) \\
 &\stackrel{\text{def}}{=} Z(r)
 \end{aligned}$$

- The lower bound on $\mathcal{H}(C_2|e = ff)$ is the entropy due to the execution of the “out of context” command, less the upper bound on the difference between the entropies due to execution “out of context” and with knowledge of the context
- $\mathcal{H}(C_2|e == ff) \geq a - \left(\frac{1}{r} - r\right) (k - \log \left(\frac{1}{r} - r\right)).$

Some inference rules for if statements

$$\frac{\Gamma \vdash B : [-,b] \quad \vdash \{\Gamma\} C_i \ x : [-,b_i] \quad i=1,2}{\vdash \{\Gamma\} \text{if } B \text{ then } C_1 \text{ else } C_2 \ x : [0, b+b_1+b_2]}$$

$$\frac{\Gamma \vdash B : [0,0] \quad \vdash \{\Gamma\} C_i \ x : [a_i, b_i]}{\vdash \{\Gamma\} \text{if } B \text{ then } C_1 \text{ else } C_2 \ x : [\min(a_1, a_2), \max(b_1, b_2)]}$$

$$\frac{\Gamma \vdash E : [-,b] \quad \vdash \{\Gamma\} C_2 \ x : [-,b_1]}{\vdash \{\Gamma\} \text{if } B \text{ then } C_1 \text{ else } C_2 \ x : [0, b+b_1+(1-r)k]} \quad 1-r = p(E = tt)$$

$$\frac{\vdash \{\Gamma\} C_2 \ x : [a, -] \quad \Gamma \vdash E_1 : [a_1, -] \quad \Gamma \vdash E_2 : [-, b_2]}{\vdash \{\Gamma\} \text{if } E_1 == E_2 \text{ then } C_1 \text{ else } C_2 \ x : [r(a-Z(r)), k]}$$

where $\frac{1}{r} - r \leq \frac{1}{2}$, $\frac{1}{2} \leq r \leq 1 - \frac{1}{2^k}$, $\mathcal{U}_k(1-r) = (a_1 - b_2)$

example

- Let P be the program

```
if (h==n) then h=0; else y=h;
if (y==m) then l=0; else l=1;
```
- assume confidential h is very unlikely to be equal to constant n
- statistically P is very similar to P':

```
if (h==m) then l=0; else l=1;
```
- for analyses of P and P' to give similar results, need analysis of first if statement of P to produce $h:[32,32]$ (assume h uniformly distributed)
- As before $h:[32, 32] \vdash (h == n):[0, \epsilon]$ where $\epsilon = \mathcal{B}(1/2^{32}) \approx 7.8 \times 10^{-9}$

example

- ❑ analysis of false branch will be $\vdash h:[32, 32] \{y=h\} y:[32, 32]$
- ❑ so get values $q = 1/2^{32}$ and $a = 32$ to use in conditional rule to deduce
- ❑ $\vdash h:[32, 32] \{C\} y:[(1 - 1/2^{32})(32 - 4.2^{-8}), 32]$ where C is the first if statement in P
- ❑ Note the bounds for y are very close to $[32, 32]$ so bounds for $\mathbf{1}$ in both P and P' will be very close
- ❑ without a good lower bound $\mathbf{1}$ will have much higher bounds in P than in P' (exercise: assume $y: [0, 32]$)

Conclusions

- ❑ analysis uses non-trivial information theoretic results in a non-trivial way.
- ❑ important to get close upper and lower bounds on leakage when leakage can be small.
- ❑ currently working on leakage rates and devising framework for leakage in interactive programs (non-deducability on strategies)