An introduction to Quantitative Information Flow (QIF)

Michele Boreale

DiSIA - Università di Firenze

IFIP WG 2.2, FCT-UNL, Lisbon
23-27 September 2013

Two elusive concepts

Confidentiality (aka Secrecy)

Sensitive information is never leaked to unintended parties. Often pursued via encryption. Protection of 'high-entropy' secrets:

- PIN's, passwords, keys, credit card numbers
- memory content
- ...

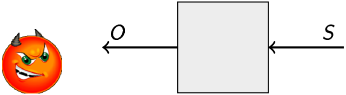
Privacy

Personal information about *individuals* is never disclosed. Often pursued via anonymization and aggregation of data. Protection of

- participation of an individual in a database
- value of an individual's sensitive (e.g. medical) attribute
- individual's purchase preferences
- ...

Attacker

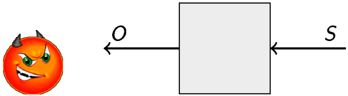
Despite a variety of concrete contexts and situations, the underlying paradigm is conceptually simple. We presuppose an **attacker** that gets to know certain **observable information** and, from this, tries her/his best to learn the **secret**.



- Attacker's task: **infer** the secret given the observable information.
- Our tasks:
  1. **quantify** the attacker's chances of success / necessary effort
  2. devise **tools and methods** to make chances as small as possible / effort as large as possible.

Attacker

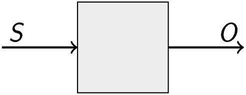
Despite a variety of concrete contexts and situations, the underlying paradigm is conceptually simple. We presuppose an **attacker** that gets to know certain **observable information** and, from this, tries her/his best to learn the **secret**.



- Attacker's task: **infer** the secret given the observable information.
- Our tasks:
  1. **quantify** the attacker's chances of success / necessary effort
  2. devise **tools and methods** to make chances as small as possible / effort as large as possible.
- **Two models**: Quantitative Information Leakage (QIF, confidentiality) and Differential Privacy (DP, privacy).

QIF: motivation and intuition

Let us consider a program/system operating taking as input a sensitive variable S and producing a public (observable) output O, as a 'black-box'.

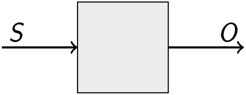


Ideal situation: Noninterference (Goguen-Meseguer 1982). Value of O does not depend on the secret S.
In practice, this is extremely hard to achieve, especially when the output O has to have some *utility*.

Example: PIN-checker

```
L=input()
if S=L then O:=yes else O:=no
```

Let us consider a program/system operating taking as input a sensitive variable S and producing a public (observable) output O, as a 'black-box'.



Ideal situation: Noninterference (Goguen-Meseguer 1982). Value of O does not depend on the secret S.
In practice, this is extremely hard to achieve, especially when the output O has to have some *utility*.

Example: PIN-checker

```
L=input()
if S=L then O:=yes else O:=no
```

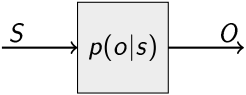
Realistic approach: measure the quantity of information (in bits) the attacker can learn about S by observing O. If this is very small - below a given threshold - decree the system *secure*.

A noisy channel model (e.g.[Chatzikokolakis, Palamidessi 2008])

(Probabilistic) programs or systems viewed as *noisy channels*:

- input S = sensitive information
- output O = observables

Noisy: fixed a given input, one can obtain different outputs each with a certain probability (probabilistic programs)

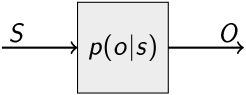


A noisy channel model (e.g.[Chatzikokolakis, Palamidessi 2008])

(Probabilistic) programs or systems viewed as *noisy channels*:

- input S = sensitive information
- output O = observables

Noisy: fixed a given input, one can obtain different outputs each with a certain probability (probabilistic programs)



Formally:

Randomization mechanism

A *randomization mechanism* is a triple $\mathcal{R} = (\mathcal{S}, \mathcal{O}, p(\cdot|\cdot))$, where:

1. $\mathcal{S}$ is a finite set of *secret inputs*, representing the sensitive information
2. $\mathcal{O}$ is a finite set of *observations*, representing the observable information
3. $p(\cdot|\cdot) \in [0,1]^{\mathcal{S}\times\mathcal{O}}$ is a *conditional probability matrix*, where each row sums up to 1.

Note: a matrix with only 0-1 entries defines an I/O function $f : \mathcal{S} \to \mathcal{O}$.

Simple examples/1

PIN-checker. Assume $0 \leq S < 4$, uniformly distributed. $\mathcal{O} = \{\mathtt{yes}, \mathtt{no}\}$.

Program:

```
\\ assume L=3
if S=L then
   O:=yes
else
   O:=no
```

Matrix:

$$p(\cdot|\cdot) = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{cc} \mathit{yes} & \mathit{no} \end{array} \\ \begin{bmatrix} 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} \end{array}$$

Simple examples/1

PIN-checker. Assume $0 \leq S < 4$, uniformly distributed. $\mathcal{O} = \{\mathtt{yes}, \mathtt{no}\}$.

Program:

```
\\ assume L=3
if S=L then
   O:=yes
else
   O:=no
```

Matrix:

$$p(\cdot|\cdot) = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{cc} \mathit{yes} & \mathit{no} \end{array} \\ \left[\begin{array}{cc} 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \end{array}\right] \end{array}$$

An interesting program (Smith '09). Assume $0 \leq S < 2^{32}$, uniformly distributed.

```
if S mod 8 = 0 then
   O := S
else
   O := 1
```

Simple examples/1

PIN-checker. Assume $0 \le S < 4$, uniformly distributed. $\mathcal{O} = \{\mathtt{yes}, \mathtt{no}\}$.

Program:

```
\\ assume L=3
if S=L then
   O:=yes
else
   O:=no
```

Matrix:

$$p(\cdot|\cdot) = \begin{array}{c c} & \begin{array}{cc} \textit{yes} & \textit{no} \end{array} \\ \begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \end{array} & \begin{bmatrix} 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} \end{array}$$

An interesting program (Smith '09). Assume $0 \le S < 2^{32}$, uniformly distributed.

```
if S mod 8 = 0 then
   O := S
else
   O := 1
```

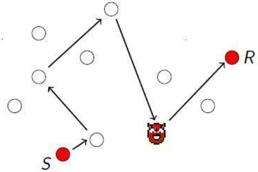
The entire secret is leaked $\frac{1}{8}$ of the times. Is it a big leak or not? We will see.

Simple examples/2

Crowds, a probabilistic anonymity protocol (Reiter, Rubin 1998).

A node is detected: is it the true sender or just a forwarder?

With three honest nodes and one corrupted, we have $\mathcal{S} = \{n_1, n_2, n_3\}$, $\mathcal{O} = \{d_1, d_2, d_3\}$ and



$$p(\cdot|\cdot) = \begin{array}{cc} & \begin{array}{ccc} d_1 & d_2 & d_3 \end{array} \\ \begin{array}{c} n_1 \\ \\ n_2 \\ \\ n_3 \end{array} & \begin{bmatrix} \frac{7}{8} & \frac{1}{16} & \frac{1}{16} \\ \\ \frac{1}{16} & \frac{7}{8} & \frac{1}{16} \\ \\ \frac{1}{16} & \frac{1}{16} & \frac{7}{8} \end{bmatrix} \end{array}$$

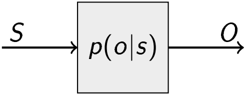
Many more examples

- in databases, queries may leak information about 'sensitive' fields
- side-channel attacks against smart-cards: exploit correlation between secret key and execution time, power consumption,...
- ...

Quantifying flow of information/1

Consider a randomization mechanism $\mathcal{R}$.

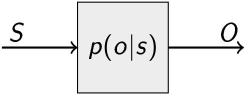
- Adversary knows prior probability distribution $p_S(\cdot)$ on $\mathcal{S}$: this also incorporates his own **background knowledge**.
- Secret and observable information form then a pair of random variables (S, O), distributed according to $p_{S,O}(s,o) = p_S(s) \cdot p(o|s)$.



Quantifying flow of information/1

Consider a randomization mechanism $\mathcal{R}$.

- Adversary knows prior probability distribution $p_S(\cdot)$ on $\mathcal{S}$: this also incorporates his own **background knowledge**.
- Secret and observable information form then a pair of random variables (S, O), distributed according to $p_{S,O}(s, o) = p_S(s) \cdot p(o|s)$.

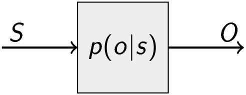


Assume we have an **uncertainty** measure $H(\cdot)$ for random variables.

Quantifying flow of information/1

Consider a randomization mechanism $\mathcal{R}$.

- Adversary knows prior probability distribution $p_S(\cdot)$ on $\mathcal{S}$: this also incorporates his own **background knowledge**.
- Secret and observable information form then a pair of random variables (S, O), distributed according to $p_{S,O}(s,o) = p_S(s) \cdot p(o|s)$.



Assume we have an **uncertainty** measure $H(\cdot)$ for random variables.

Information flow = reduction in uncertainty

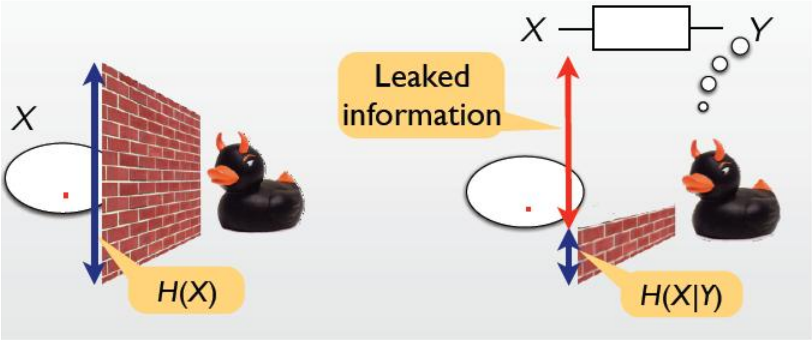
$$
\begin{aligned}
\text{Information Flow} &= \text{prior uncertainty - posterior uncertainty} \\
&\stackrel{\text{def}}{=} H(S) - H(S|O)
\end{aligned}
$$

Quantifying flow of information/2

Note:

- $I(S;O) \stackrel{\text{def}}{=} H(S) - H(S|O)$ is often named *mutual information* in Information Theory.
- $H(S|O)$ represents *average* posterior uncertainty. E.g. $\sum_o p(o)H(S|O=o)$.
- The flow is 0 precisely when S and O are independent: only in this case $H(S|O) = H(S)$, hence $I(S;O) = 0$ (Non-Interference).
- If $H(S) \approx 0$, then $I(S;O) \approx 0$.
  Alas, there is little we can do if passwords are badly chosen!

Quantifying flow of information - pictorially



Uncertainty $H(X)$ captures "height of wall", in terms of chances of success of guessing, or expected effort for learning, the secret X.

(courtesy of Boris Köpf)

But what is 'Uncertainty'?

Several proposals for $H(\cdot)$. First (obvious) attempt:

Shannon entropy (Shannon 1948)

$$\begin{aligned} H_{\mathrm{Sh}}(S) &\stackrel{\mathrm{def}}{=} -\sum_s p(s)\log p(s) \\ &= \text{Average n. of binary questions necessary to learn } S \end{aligned}$$

But what is 'Uncertainty'?

Several proposals for $H(\cdot)$. First (obvious) attempt:

Shannon entropy (Shannon 1948)

$$\begin{aligned} H_{\mathrm{Sh}}(S) &\stackrel{\mathrm{def}}{=} -\textstyle\sum_s p(s)\log p(s) \\ &= \text{Average n. of binary questions necessary to learn } S \end{aligned}$$

PIN-checking example. Let S be a 5-digits PIN, chosen at random.

```
if S=L then O:=yes else O:=no
```

- Prior uncertainty: $H(S) = \log 10^5 \approx 16.6096$ bits
- Posterior uncertainty.
  - $H(S|O =' yes') = 0$
  - $H(S|O =' no') = \log(10^5 - 1)$

  On average: $H(S|O) = (\frac{10^5-1}{10^5})\log(10^5 - 1) \approx 16.6094$
- Information flow = H(S)-H(S|O) ≈ 0.0002 bits

So my PIN is safe, after all...

Does Shannon entropy properly reflect 'how difficult' is to guess?

Assume $0 \leq S < 2^{32}$, uniformly distributed.

```
if S mod 8 = 0 then
   O := S
else
   O := 1
```

- Prior uncertainty: $H(S) = 32$ bits
- Posterior uncertainty.
  - $H(S|O=y) = 0$ for $y \neq 1$, happens $\frac{1}{8}$ of the times;
  - $H(S|O=1) = 32 - \log\frac{8}{7}$, happens $\frac{7}{8}$ of the times;

  On average: $H(S|O) = \frac{7}{8} \times (32 - \log\frac{8}{7}) \approx 28 - 0.169$ bits
- Information flow = H(S)-H(S|O) $\approx$ 4.169 bits

Does Shannon entropy properly reflect 'how difficult' is to guess?

Assume $0 \leq S < 2^{32}$, uniformly distributed.

```
if S mod 8 = 0 then
   O := S
else
   O := 1
```

- Prior uncertainty: $H(S) = 32$ bits
- Posterior uncertainty.
  - $H(S|O = y) = 0$ for $y \neq 1$, happens $\frac{1}{8}$ of the times;
  - $H(S|O = 1) = 32 - \log \frac{8}{7}$, happens $\frac{7}{8}$ of the times;

  On average: $H(S|O) = \frac{7}{8} \times (32 - \log \frac{8}{7}) \approx 28 - 0.169$ bits
- Information flow = H(S)-H(S|O) ≈ 4.169 bits

Suggests that $\approx 7/8$ of the secret bits remain unleaked. However, adversary can guess the **whole** 32 bits of the secret $\frac{1}{8}$ of the times!

A more operational notion

Min-entropy (Renyi 1961)

$$\begin{aligned} H_\infty(S) &\stackrel{\text{def}}{=} -\log \max_s p(s) \\ &= -\log \left(\text{chances of successfully guess } S \text{ in one try}\right) \end{aligned}$$

A more operational notion

Min-entropy (Renyi 1961)

$$
\begin{aligned}
H_\infty(S) &\stackrel{\text{def}}{=} -\log \max_s p(s) \\
&= -\log\left(\text{chances of successfully guess } S \text{ in one try}\right)
\end{aligned}
$$

$$
H_\infty(S|O) \stackrel{\text{def}}{=} -\log \Big(\underbrace{\sum_o p(o) \max_s p(s|o)}_{=\text{avg. a posteriori chances of success}} \Big)
$$

- Proposed by Smith in 2009 as an alternative to Shannon for QIF
- Clear operational significance:

$$
Leakage = H(S) - H(S|O) = \log \frac{p(\text{success a posteriori })}{p(\text{success a priori})}
$$

1 bit gained by attacker = success probability doubled!

Some results/1

For a **deterministic** program and a **uniform prior**

$$\text{Leakage} = \log(\#\text{ distinct output values of the program })$$

(Smith 2009) In other words, leakage only depends on $|\mathrm{Im}(f)|$, where $f : \mathcal{S} \to \mathcal{O}$ (termination considered observable).

Some results/1

For a **deterministic** program and a **uniform prior**

$$\text{Leakage} = \log(\#\text{ distinct output values of the program })$$

(Smith 2009) In other words, leakage only depends on $|\mathrm{Im}(f)|$, where $f : \mathcal{S} \to \mathcal{O}$ (termination considered observable).

Example:

```
if S mod 8 = 0 then
   O := S
else
   O := 1
```

This leaks $\log 2^{29} = 29$ bits of min-entropy (vs. ≈ 4 of Shannon) about S.

Some results/1

For a **deterministic** program and a **uniform prior**

$$\text{Leakage} = \log(\#\text{ distinct output values of the program })$$

(Smith 2009) In other words, leakage only depends on $|\mathrm{Im}(f)|$, where $f : \mathcal{S} \to \mathcal{O}$ (termination considered observable).

Example:

```
if S mod 8 = 0 then
   O := S
else
   O := 1
```

This leaks $\log 2^{29} = 29$ bits of min-entropy (vs. ≈ 4 of Shannon) about S.

Proof:

$$\begin{aligned}
H(S|O) &= -\log\left(\textstyle\sum_{o:p(o)>o} p(o)\max_s p(s|o)\right) \\
&= -\log\left(\textstyle\sum_{o:p(o)>o} p(o)\max_s \frac{p(o|s)p(s)}{p(o)}\right) && \text{(Bayes)} \\
&= -\log\left(\textstyle\sum_{o:p(o)>o} \max_s p(o|s)p(s)\right) \\
&= -\log\left(\frac{1}{|\mathcal{S}|}\textstyle\sum_{o:p(o)>o} \max_s p(o|s)\right) && \text{(uniform prior)} \\
&= -\log\left(\frac{1}{|\mathcal{S}|}\textstyle\sum_{o:p(o)>o} 1\right) && \text{(determinism)} \\
&= -\log\left(\frac{|\mathrm{Im}(f)|}{|\mathcal{S}|}\right) \\
H(S) - H(S|O) &= -\log\frac{1}{|\mathcal{S}|} + \log\frac{|\mathrm{Im}(f)|}{|\mathcal{S}|} = \log\left(|\mathcal{S}|\frac{|\mathrm{Im}(f)|}{|\mathcal{S}|}\right) \\
&= \log|\mathrm{Im}(f)|
\end{aligned}$$

Some results/2

For a general **probabilistic** program and **repeated observations**; conditional independence of $O_1, ..., O_n$ given S is typically assumed:

$$p(o_1, ..., o_n|s) = \Pi_j p(o_j|s)$$

$Leakage(n) \stackrel{\mathrm{def}}{=} I(S; O^n)$. Under a uniform prior, as $n \to +\infty$

$$Leakage(n) \to \log(\#\ \text{distinct } \textit{indistinguishability} \text{ classes of the program})$$

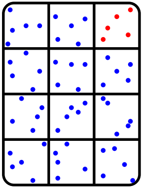
((Boreale et al. 2011) for H_∞; for generic uncertainty measures, (Boreale and Pampaloni 2013). Note: exact rate of convergence can be determined from the matrix.)

What does that mean?

Indistinguishability

Given $s, s' \in \mathcal{S}$, we let $s \equiv s'$ iff for each o: $p(o|s) = p(o|s')$.
This means rows s and s' in matrix $p(\cdot|\cdot)$ are equal.

Intuition: with infinitely many observations, precisely the *indistinguishability class* of the secret will be learned by the attacker.



In the case of uniform prior distribution, $I(S; [S]_\equiv) = \log K$, where K is the number of classes.

Example:

```
if S mod 8 = rnd[0..7] then
   O := S
else
   O := 1
```

In this case, $K = |\mathcal{S}| = 2^{32}$, hence asymptotic leakage is 32 bits.

Further research/1: Compositionality

- *Non-expansiveness* for sequential and parallel composition (Köpf et al., Smith et al. 2012), also in a process-algebraic setting (Boreale 2006):

$$Leakage(P_1 \circ P_2) \leq Leakage(P_1) + Leakage(P_2)$$

Further research/1: Compositionality

- *Non-expansiveness* for sequential and parallel composition (Köpf et al., Smith et al. 2012), also in a process-algebraic setting (Boreale 2006):

$$Leakage(P_1 \circ P_2) \leq Leakage(P_1) + Leakage(P_2)$$

- In the case of Shannon entropy:
  - Leakage $= I(S;O) = I(O;S) = H(O) - \underbrace{H(O|S)}_{=0,\text{ if } P \text{ det.}} = H(O)$
  - Chain rule (provided ϕ depends only on O):

    $$H(O) = H(\phi) + H(O|\phi)$$

  - Hence for if-then-else

    $$Leakage(\texttt{if b then c1 else c2}) = H(\texttt{b}) + p(\texttt{b})H(\texttt{c1}|\texttt{b}) + p(\neg\texttt{b})H(\texttt{c2}|\neg\texttt{b})$$

    (provided final value of `O` determines initial value of `b`.)
  - Extensible to looping constructs, cf. Malacaria, POPL'07.

Example (if-then-else)

$$\begin{aligned} \mathit{Leakage}(\texttt{if b then c1 else c2}) \;=\; & H(\texttt{b}) + p(\texttt{b})H(\texttt{c1}|\texttt{b}) \\ & + p(\neg\texttt{b})H(\texttt{c2}|\neg\texttt{b}) \end{aligned}$$

```
if S%8 =0 then
   O := S
else
   O := 1
```

$$\begin{aligned} \mathit{Leakage} &= H(\texttt{S\% =0}) + p(\texttt{S\%8 =0})H(\texttt{O:=S}|\texttt{S\%8 =0}) \\ &\qquad + p(\texttt{S\%8 !=0})H(\texttt{O:=1}|\texttt{S\%8 !=0}) \\ &= H(\tfrac{1}{8},\tfrac{7}{8}) + \tfrac{1}{8}\times 29 \\ &\qquad + \tfrac{7}{8}\times 0 \\ &\approx 4.169 \end{aligned}$$

Further research/2

- **Trace-based observations?** Systems go through several states before producing a result, if any. At each step, attacker detects a (noisy) observation of the current state, like in *Hidden Markov Models*.
  Observation = trace, hence set of observables is now $\mathcal{O}^*$.
  Much of the theory extends smoothly (Boreale et al. 2011).

$$\begin{tikzcd} S_1 \arrow[r] \arrow[d] & S_2 \arrow[r] \arrow[d] & S_3 \arrow[r] \arrow[d] & \cdots \\ O_1 & O_2 & O_3 & \cdots \end{tikzcd}$$

Further research/2

- **Trace-based observations?** Systems go through several states before producing a result, if any. At each step, attacker detects a (noisy) observation of the current state, like in *Hidden Markov Models*.
  Observation = trace, hence set of observables is now $\mathcal{O}^*$.
  Much of the theory extends smoothly (Boreale et al. 2011).

$$\begin{tikzcd} S_1 \arrow[r] \arrow[d] & S_2 \arrow[r] \arrow[d] & S_3 \arrow[r] \arrow[d] & \cdots \\ O_1 & O_2 & O_3 & \cdots \end{tikzcd}$$

- **Adaptive attackers?** $O = f(S, q)$, for *query* $q \in \mathcal{Q}$. Attacker can repeatedly choose and submit queries q, based on previous observations, hence play a *strategy* $\sigma : O^* \to \mathcal{Q}$. Complete observation is O_σ. Leakage is

$$H(S) - \inf_{\sigma} H(S|O_\sigma)$$

  Optimal strategy computable via *MDP*-based algorithms. Non-adaptive, brute force strategies are as efficient as adaptive ones, up to a length expansion of $\times|\mathcal{Q}|$. (Boreale, Pampaloni 2013).

Further research/3

- **Relation with privacy**. Aim: protect information about any *individual* in a DB, independently of attacker's prior knowledge. Ideally, even *participation* of the individual in the DB should be hidden.

- QIF may not be adequate, because it is an *average* measure

$$\begin{array}{c c} & \begin{array}{cc} <2.20 & \geq 2.20 \end{array} \\ \begin{array}{c} i_1 \\ i_2 \\ i_3 \\ i_4 \\ i_5 \\ \vdots \\ i_{10^9} \end{array} & \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ \vdots & \vdots \\ 1 & 0 \end{bmatrix} \end{array}$$

Individual i_1 is the only one with height ≥ 2.20 m.
Yet (min-entropy): $I(S;O) = H(S) - H(S|O) = 1$ bit, out of 30 bits.

Further research/3

- **Relation with privacy**. Aim: protect information about any *individual* in a DB, independently of attacker's prior knowledge. Ideally, even *participation* of the individual in the DB should be hidden.

- QIF may not be adequate, because it is an *average* measure

$$
\begin{array}{c|cc|}
 & <2.20 & \geq 2.20 \\
i_1 & 0 & 1 \\
i_2 & 1 & 0 \\
i_3 & 1 & 0 \\
i_4 & 1 & 0 \\
i_5 & 1 & 0 \\
\vdots & \vdots & \vdots \\
i_{10^9} & 1 & 0
\end{array}
$$

  Individual i_1 is the only one with height ≥ 2.20 m.
  Yet (min-entropy): $I(S;O) = H(S) - H(S|O) = 1$ bit, out of 30 bits.

- Also, answers of the mechanism should not be deterministic. E.g. query gives exact *average height*: attacker could make a query *before* and *after* insertion of individual i, and learn i's height.

Differential Privacy: definitions

Definition (Dwork 2006). Let $\epsilon > 0$, assume $\mathcal{S}$ is a set of DB instances. A randomization mechanism is ϵ-*differentially private* if for any two DB instances s and s' which differ by exactly *one* individual, for each $o \in \mathcal{O}$:

$$2^{-\epsilon} \leq \frac{p(o|s)}{p(o|s')} \leq 2^{\epsilon}$$

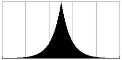
Differential Privacy: definitions

Definition (Dwork 2006). Let $\epsilon > 0$, assume $\mathcal{S}$ is a set of DB instances. A randomization mechanism is ϵ-*differentially private* if for any two DB instances s and s' which differ by exactly *one* individual, for each $o \in \mathcal{O}$:

$$2^{-\epsilon} \leq \frac{p(o|s)}{p(o|s')} \leq 2^{\epsilon}$$

Laplacian noise. Let $Q : \mathcal{S} \to \mathbb{R}$ be a query function. Let $\Delta = \max_{s \text{ adj. } s'} |Q(s) - Q(s')|$ be the *sensitivity* of Q (e.g., if Q is the counting query, $\Delta = 1$). The mechanism defined by

$$O = Q(S) + Y \text{ where } Y \sim \frac{2^{-|y|\frac{\epsilon}{\Delta}}}{Z}$$



is differentially private, whatever S.

Conclusion

- QIF: a model of confidentiality based on simple information-theoretic concepts
- Very active research area in Theoretical Computer Science. Strong relations with Differential Privacy and Data Base communities.
- Challenges:
  1. Incorporate QIF concepts and analysis in programming languages (type systems, tools,....). Promising work by Köpf and Rybalchenko on automated estimation of QIF; for DP, cf. McSherry's PINQ.
  2. Real-world applications. Promising work on CPU caches and timing leaks in RSA (cf. work by Köpf and Smith).

Some papers (personal take) / 1

- M. S. Alvim, M. E. Andrés, C. Palamidessi. Quantitative information flow in interactive systems. *Journal of Computer Security* 20(1): 3-50 (2012)
- M. Backes, B. Köpf. Formally Bounding the Side-Channel Leakage in Unknown-Message Attacks. *ESORICS 2008*: 517-532
- M. Backes, B. Köpf, A. Rybalchenko. Automatic Discovery and Quantification of Information Leaks. *IEEE Symposium on Security and Privacy* 2009: 141-153
- M. Boreale. Quantifying information leakage in process calculi. *Inf. Comput.* 207(6): 699-725 (2009)
- M. Boreale, F. Pampaloni, M. Paolini. Quantitative Information Flow, with a View. *ESORICS* 2011: 588-606
- M. Boreale, F. Pampaloni, M. Paolini. Asymptotic Information Leakage under One-Try Attacks. *FOSSACS* 2011: 396-410 (full version to appear in *MSCS*).
- M. Boreale, F. Pampaloni. On the limits of adaptive adversaries. Submitted, 2013.

Some papers (personal take) / 2

- K. Chatzikokolakis, C. Palamidessi, P. Panangaden. Anonymity protocols as noisy channels. *Inf. Comput.* 206(2-4): 378-401 (2008)
- D. Clark, S. Hunt, P. Malacaria. Quantitative Analysis of the Leakage of Confidential Data. *Electr. Notes Theor. Comput. Sci.* 59(3): 238-251 (2001)
- J. Heusser, P. Malacaria. Quantifying information leaks in software. *ACSAC* 2010: 261-269
- B. Köpf, L. Mauborgne, M. Ochoa. Automatic Quantification of Cache Side-Channels. *CAV* 2012: 564-580
- B. Köpf, G. Smith. Vulnerability Bounds and Leakage Resilience of Blinded Cryptography under Timing Attacks. *CSF* 2010: 44-56
- P. Malacaria. Assessing security threats of looping constructs. *POPL 2007*: 225-235
- G. Smith. On the Foundations of Quantitative Information Flow. *FOSSACS 2009*: 288-302