Quantification of Information Leakage by *Statistical Method* Combined with *Program Analysis*

Yusuke Kawamoto
*(AIST, Japan)*

Joint work with
- Fabrizio Biondi *(INRIA/IRISA Rennes, France)*
- Axel Legay *(INRIA/IRISA Rennes, France)*

In French-Japanese Cybersecurity Workshop, September 2016
Information leakage

In real world systems, there are usually some information leakage.

Receipt of credit card
always leaks 4 digits of card number

****  ****  ****   1234
Acceptable

Timing/power consumption in decrypting ciphertexts
leaks information on the secret key

Security breach !

Password checking
leaks the password with small probability

\[
\text{in (guess)} \\
\text{if } guess == password \\
\text{then out(secret)}
\]
Acceptable

Message length/packet patterns
leaks the sender of the message

Privacy breach !

Want to quantify the information leakage !
Entropy as a secrecy measure

Larger entropy $\Leftrightarrow$ Larger uncertainty $\Leftrightarrow$ Stronger secrecy

- When a secret value *uniformly* distributed over $\{1, 2, \ldots, 2^{999}\}$, it’s hard to guess the secret.

  
  \[
  \begin{array}{c|c|c|c|c}
  sec & 1 & 2 & 3 & \ldots & 2^{999} \\
  \hline
  2^{-999} & 2^{-999} & 2^{-999} & \ldots & 2^{-999} \\
  \end{array}
  \]

  Shannon entropy $= 999$ bit ($= 2^{999} \cdot (–2^{-999} \cdot \log 2^{-999})$)

- When a secret value is *not* uniformly distributed, it’s easier to guess the secret.

  
  \[
  \begin{array}{c|c|c|c|c}
  sec & 1 & 2 & 3 & \ldots & 2^{999} \\
  \hline
  0.7 & 0.3 & 0 & \ldots & 0 \\
  \end{array}
  \]

  Shannon entropy $= 0.88$ bit ($= –(0.7 \log 0.7) – (0.3 \log 0.3)$)
Overview of this talk

(1) Introduction on modelling information leakage:
   How much secret information is leaked by output in a system?

(2) **Statistical method** for estimating information leakage amount:
   The accuracy of estimation is described by 95% confidence intervals.

(3) **Hybrid method** combining **statistical method** with **program analysis**
   improves the quality and cost of analysis.
Information flow

Systems are modeled as information-theoretic channels:

- Secret $s$ is input to the system.
- Output $o$ is the result.

Examples:
- Secret decryption key
- Sender of message
- Secret in a process
- Anonymity protocol
- Cipher
- Decryption time & power consumption
- Packet patterns
- Scheduling of processes
- Process scheduler
Modelling a probabilistic system

Probabilistic system

Initial state

s = 0

s = 1

s = 2

s = 3

Observed outputs

(\textit{no} non-deterministic transitions)

Joint distribution of secret & output

<table>
<thead>
<tr>
<th>Secret</th>
<th>Output</th>
<th>o = 0</th>
<th>o = 1</th>
<th>o = 2</th>
<th>o = 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>s = 0</td>
<td>0.125</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.125</td>
</tr>
<tr>
<td>s = 1</td>
<td>0</td>
<td>0.125</td>
<td>0.125</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>s = 2</td>
<td>0</td>
<td>0.125</td>
<td>0.125</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>s = 3</td>
<td>0.125</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.125</td>
</tr>
</tbody>
</table>

1 bit of s is leaked when observing \( o = 1 \)
Modelling information leakage

- Probabilities of secrets

Before observing output:

\[ p(s) \]  Distribution on secrets

<table>
<thead>
<tr>
<th>Secret</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s = 0 )</td>
<td>0.25</td>
</tr>
<tr>
<td>( s = 1 )</td>
<td>0.25</td>
</tr>
<tr>
<td>( s = 2 )</td>
<td>0.25</td>
</tr>
<tr>
<td>( s = 3 )</td>
<td>0.25</td>
</tr>
</tbody>
</table>

After observing output:

\[ p(s, o) \]  Joint distribution on secrets & outputs

<table>
<thead>
<tr>
<th>Output Secret</th>
<th>( o = 0 )</th>
<th>( o = 1 )</th>
<th>( o = 2 )</th>
<th>( o = 3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s = 0 )</td>
<td>0.125</td>
<td>0</td>
<td>0</td>
<td>0.125</td>
</tr>
<tr>
<td>( s = 1 )</td>
<td>0</td>
<td>0.125</td>
<td>0.125</td>
<td>0</td>
</tr>
<tr>
<td>( s = 2 )</td>
<td>0</td>
<td>0.125</td>
<td>0.125</td>
<td>0</td>
</tr>
<tr>
<td>( s = 3 )</td>
<td>0.125</td>
<td>0</td>
<td>0</td>
<td>0.125</td>
</tr>
</tbody>
</table>
Modelling information leakage

- Probabilities of secrets
  Before observing output:
  \[ p(s) \] Distribution on secrets

<table>
<thead>
<tr>
<th>Secret</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>s = 0</td>
<td>0.25</td>
</tr>
<tr>
<td>s = 1</td>
<td>0.25</td>
</tr>
<tr>
<td>s = 2</td>
<td>0.25</td>
</tr>
<tr>
<td>s = 3</td>
<td>0.25</td>
</tr>
</tbody>
</table>

After observing output:
\[ p(s, o) \] Joint distribution on secrets & outputs

<table>
<thead>
<tr>
<th>Secret</th>
<th>o = 0</th>
<th>o = 1</th>
<th>o = 2</th>
<th>o = 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>s = 0</td>
<td>0.125</td>
<td>0</td>
<td>0</td>
<td>0.125</td>
</tr>
<tr>
<td>s = 1</td>
<td>0</td>
<td>0.125</td>
<td>0.125</td>
<td>0</td>
</tr>
<tr>
<td>s = 2</td>
<td>0</td>
<td>0.125</td>
<td>0.125</td>
<td>0</td>
</tr>
<tr>
<td>s = 3</td>
<td>0.125</td>
<td>0</td>
<td>0</td>
<td>0.125</td>
</tr>
</tbody>
</table>

- Amount of information leakage
  \[ \text{(Leakage amount)} = (\text{entropy before observation}) - (\text{entropy after observation}) \]

  E.g.
  \[ \text{(Mutual information)} = - \sum_{s \in \text{Sec}} p(s) \log p(s) - \sum_{o \in \text{Obs}} p(o) \sum_{s \in \text{Sec}} p(s|o) \log p(s|o) \]

  Remark: Other kinds of entropy (e.g. min-entropy) for other attack scenarios
Modelling information leakage

- Probabilities of secrets
  
  Before observing output:
  
  $p(s)$ Distribution on secrets
  
<table>
<thead>
<tr>
<th>Secret</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s = 0$</td>
<td>0.25</td>
</tr>
<tr>
<td>$s = 1$</td>
<td>0.25</td>
</tr>
<tr>
<td>$s = 2$</td>
<td>0.25</td>
</tr>
<tr>
<td>$s = 3$</td>
<td>0.25</td>
</tr>
</tbody>
</table>
  
  After observing output:
  
  $p(s, o)$ Joint distribution on secrets & outputs
  
<table>
<thead>
<tr>
<th>Secret</th>
<th>$o = 0$</th>
<th>$o = 1$</th>
<th>$o = 2$</th>
<th>$o = 3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s = 0$</td>
<td>0.125</td>
<td>0</td>
<td>0</td>
<td>0.125</td>
</tr>
<tr>
<td>$s = 1$</td>
<td>0</td>
<td>0.125</td>
<td>0.125</td>
<td>0</td>
</tr>
<tr>
<td>$s = 2$</td>
<td>0</td>
<td>0.125</td>
<td>0.125</td>
<td>0</td>
</tr>
<tr>
<td>$s = 3$</td>
<td>0.125</td>
<td>0</td>
<td>0</td>
<td>0.125</td>
</tr>
</tbody>
</table>

- Amount of information leakage
  
  (Leakage amount) = (entropy before observation) – (entropy after observation)

- Hardness in computing leakage amount
  
  - Computation is hard if there are many traces, shown by T. Terauchi (JAIST, Japan).
  - Limited compositionality (e.g. in QEST’14 with K. Chatzikokolakis & C. Palamidessi):
    Composing two systems may produce additional information leakage.
Overview

- Model of information leakage
- Statistical method for estimating leakage amount
- Hybrid statistical method
- Evaluation of the method
- Summary & future work
Statistical method

[Chatzikokolakis, Chothia, Guha’10]
CSF’13, ESORICS’14 with T. Chothia, C. Novakovic, D. Parker

Tool LeakWatch

System

Execution traces

Running many times

Numbers of traces

Approximate probability distribution

Estimated leakage: 1.007 bit?

Record of 1000 traces

(bool) \( sec_0 := \{ 0 \rightarrow 0.5, 1 \rightarrow 0.5 \} \);
(bool) \( sec_1 := \{ 0 \rightarrow 0.5, 1 \rightarrow 0.5 \} \);
(secret) \( sec_0, sec_1 \);
(bool) \( r := \{ 0 \rightarrow 0.5, 1 \rightarrow 0.5 \} \);
(bool) \( obs_0 := sec_0 \ XOR r \);
(bool) \( obs_1 := sec_1 \ XOR r \);
(output) \( obs_0, obs_1 \);

Probabilistic assignment produces different traces.
Statistical method [Chatzikokolakis, Chothia, Guha’10] CSF’13, ESORICS’14 with T. Chothia, C. Novakovic, D. Parker

- Tool LeakWatch
- System
  - Execution traces
    - Running many times
  - Numbers of traces
    - Approximate probability distribution
      - Estimated leakage: 1.007 bit

bool \( sec_0 \) := \{ 0 \rightarrow 0.5, 1 \rightarrow 0.5 \};
bool \( sec_1 \) := \{ 0 \rightarrow 0.5, 1 \rightarrow 0.5 \};
bool \( r \) := \{ 0 \rightarrow 0.5, 1 \rightarrow 0.5 \};
bool \( obs_0 \) := \( sec_0 \) XOR \( r \);
bool \( obs_1 \) := \( sec_1 \) XOR \( r \);
output \( obs_0 \), \( obs_1 \);

Probabilistic assignment produces different traces.

Record of 1000 traces:

<table>
<thead>
<tr>
<th>secret</th>
<th>output</th>
<th>( obs_0=0 )</th>
<th>( obs_0=1 )</th>
<th>( obs_1=0 )</th>
<th>( obs_1=1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( sec_0=0, sec_1=0 )</td>
<td>121</td>
<td>0</td>
<td>0</td>
<td>129</td>
<td></td>
</tr>
<tr>
<td>( sec_0=0, sec_1=1 )</td>
<td>0</td>
<td>136</td>
<td>114</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>( sec_0=1, sec_1=0 )</td>
<td>0</td>
<td>132</td>
<td>118</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>( sec_0=1, sec_1=1 )</td>
<td>117</td>
<td>0</td>
<td>0</td>
<td>133</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>secret</th>
<th>output</th>
<th>( obs_0=0 )</th>
<th>( obs_0=1 )</th>
<th>( obs_1=0 )</th>
<th>( obs_1=1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( sec_0=0, sec_1=0 )</td>
<td>0.121</td>
<td>0</td>
<td>0</td>
<td>0.129</td>
<td></td>
</tr>
<tr>
<td>( sec_0=0, sec_1=1 )</td>
<td>0</td>
<td>0.136</td>
<td>0.114</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>( sec_0=1, sec_1=0 )</td>
<td>0</td>
<td>0.132</td>
<td>0.118</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>( sec_0=1, sec_1=1 )</td>
<td>0.117</td>
<td>0</td>
<td>0</td>
<td>0.133</td>
<td></td>
</tr>
</tbody>
</table>
**Statistical method**

[Chatzikokolakis, Chothia, Guha’10]
CSF’13, ESORICS’14 with T. Chothia, C. Novakovic, D. Parker

**Tool LeakWatch**

1. System
   - Running many times

2. Execution traces

3. Numbers of traces

4. Approximate probability distribution

**Estimated leakage:**
1.007 bit?
1.001 bit

95% confidence interval:
[0.998, 1.003]

**Approximate distribution**

<table>
<thead>
<tr>
<th>secret</th>
<th>output</th>
<th>obs0=0</th>
<th>obs0=0</th>
<th>obs0=1</th>
<th>obs0=1</th>
</tr>
</thead>
<tbody>
<tr>
<td>sec0=0, sec1=0</td>
<td>0.125</td>
<td>0</td>
<td>0</td>
<td>0.125</td>
<td></td>
</tr>
<tr>
<td>sec0=0, sec1=1</td>
<td>0</td>
<td>0.125</td>
<td>0.125</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>sec0=1, sec1=0</td>
<td>0</td>
<td>0.125</td>
<td>0.125</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>sec0=1, sec1=1</td>
<td>0.125</td>
<td>0</td>
<td>0</td>
<td>0.125</td>
<td></td>
</tr>
</tbody>
</table>

**True distribution**

<table>
<thead>
<tr>
<th>secret</th>
<th>output</th>
<th>obs0=0</th>
<th>obs0=0</th>
<th>obs0=1</th>
<th>obs0=1</th>
</tr>
</thead>
<tbody>
<tr>
<td>sec0=0, sec1=0</td>
<td>0.125</td>
<td>0</td>
<td>0</td>
<td>0.125</td>
<td></td>
</tr>
<tr>
<td>sec0=0, sec1=1</td>
<td>0</td>
<td>0.125</td>
<td>0.125</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>sec0=1, sec1=0</td>
<td>0</td>
<td>0.125</td>
<td>0.125</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>sec0=1, sec1=1</td>
<td>0.125</td>
<td>0</td>
<td>0</td>
<td>0.125</td>
<td></td>
</tr>
</tbody>
</table>

**True value**
1.000 bit

**Estimate**
1.007 bit

By Statistics
Statistical method

[Chatzikokolakis, Chothia, Guha’10]
CSF’13, ESORICS’14 with T. Chothia, C. Novakovic, D. Parker

Tool LeakWatch

1. Running many times
2. Execution traces
3. Numbers of traces
4. Approximate probability distribution

Approximate distribution

<table>
<thead>
<tr>
<th>secret</th>
<th>output</th>
<th>obs0=0</th>
<th>obs0=1</th>
<th>obs1=0</th>
<th>obs1=1</th>
</tr>
</thead>
<tbody>
<tr>
<td>sec0=0, sec1=0</td>
<td>0.121</td>
<td>0</td>
<td>0</td>
<td>0.129</td>
<td></td>
</tr>
<tr>
<td>sec0=0, sec1=1</td>
<td>0</td>
<td>0.136</td>
<td>0.114</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>sec0=1, sec1=0</td>
<td>0</td>
<td>0.132</td>
<td>0.118</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>sec0=1, sec1=1</td>
<td>0.117</td>
<td>0</td>
<td>0</td>
<td>0.133</td>
<td></td>
</tr>
</tbody>
</table>

By Statistics
Estimated leakage: 1.007 bit?
1.001 bit
95% confidence interval: [0.998, 1.003]

More traces, more accurate estimate
(less bias and smaller confidence interval).
Statistical method

[Chatzikokolakis, Chothia, Guha’10]
CSF’13, ESORICS’14 with T. Chothia, C. Novakovic, D. Parker

Implementation is enough. Source code is not necessary.

May include side-channel info e.g. timing

Running many times
E.g. Pseudorandom number generators

- Consecutive pseudorandom numbers $r_1$, $r_2$ (by a stateful generator) are correlated.
- By using LeakWatch, we can estimate how much information on $r_1$ is leaked by $r_2$.

![Graph showing leakage estimates and distinguishability](image-url)
Overview

• Model of information leakage
• Statistical method for estimating leakage amount
• Hybrid statistical method
• Evaluation of the method
• Summary & future work
Statistical methods vs formal methods

• Both methods have advantages and disadvantages:

<table>
<thead>
<tr>
<th></th>
<th>Statistical methods (Monte Carlo simulation)</th>
<th>Formal methods (program analysis)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis approach</td>
<td>Black box (analyzing execution traces)</td>
<td>White box (analyzing a source code)</td>
</tr>
<tr>
<td>Produces</td>
<td>Estimate + confidence interval</td>
<td>Exact value</td>
</tr>
<tr>
<td>Reduces costs by</td>
<td>Random sampling</td>
<td>Knowledge of code &amp; abstraction</td>
</tr>
<tr>
<td>Impractical for</td>
<td>Large joint distribution matrix</td>
<td>Large number of traces</td>
</tr>
</tbody>
</table>

- E.g. Linear congruential generators
  
  \[ r_2 := (4801 \times r_1 + 83) \mod 8192 \]

  Many internal states

- E.g. Conditional branching
  
  \[
  \begin{align*}
  & \text{if } 5 \leq \text{sec} \leq 100 \\
  & \text{then } \text{obs} := 1; \\
  & \text{else } \ldots .
  \end{align*}
  \]

  We do not have to check every value of \text{sec}.
Motivation

- Examples that **neither statistical method nor program analysis** can handle:

![Diagram showing a complicated subsystem with two branches: one with a subsystem with too many traces and the other with a subsystem with too large matrix. The initial state has a probability of 0.5 for each branch.](image)

- Combine the two methods!
Motivation

- Examples that **neither statistical method nor program analysis** can handle:

---

**Complicated subsystem**

- Initial state
  - Probability 0.5

Subsystems:
- Subsystem with **too many traces**
- Subsystem with **too large matrix**

**Statistical method**

**Program analysis**

---

**Leakage by rare events**

- Initial state
  - Probability 0.01

States:
- No leakage
- Much leakage

- Too many traces are executed
- Too few traces are executed

Sample sizes:
- Smaller sample size
- Larger sample size

**Combine the two methods!**

**Kind of importance sampling!**

---

May not be sampled
Hybrid statistical analysis

• We combine statistical method with program analysis.
Hybrid statistical analysis

• We combine statistical method with program analysis.

Example of composing 3 subsystems
Hybrid statistical analysis

- We combine statistical method with program analysis.

\[
\text{Estimated distribution} = \text{Precise sub-distribution} + \text{Approximate sub-distribution} + \text{Approximate sub-distribution} + \text{Precise sub-distribution}
\]

- bias = \( \frac{b_1}{n_1} + \frac{b_2}{n_2} \)
- variance = \( \frac{\nu_1}{n_1} + \frac{\nu_2}{n_2} \)

We ignored higher order terms in Taylor expansion of mutual information.
Hybrid statistical analysis

• Formally… please see our paper

Theorem

**expectation**

\[
E(\hat{I}(X; Y)) = I(X; Y) + \sum_{i \in \mathcal{I}} \frac{\theta_i^2}{2n_i} \left( \sum_{(x, y) \in \mathcal{D}} \phi_{ixy} - \sum_{x \in \mathcal{X}^+} \phi_{ix} - \sum_{y \in \mathcal{Y}^+} \phi_{iy} \right) + \mathcal{O}(n_i^{-2})
\]

where \( \phi_{ixy} = \frac{D_i[x, y] - D_i[x, y]^2}{P_{XY}[x, y]} \), \( \phi_{ix} = \frac{D_{Xi}[x] - D_{Xi}[x]^2}{P_X[x]} \) and \( \phi_{iy} = \frac{D_{Yi}[y] - D_{Yi}[y]^2}{P_Y[y]} \).

**variance**

\[
V(\hat{I}(X; Y)) = \sum_{i \in \mathcal{I}} \frac{\theta_i^2}{n_i} \left( \sum_{(x, y) \in \mathcal{D}} D_i[x, y] \left(1 + \log \frac{P_X[x]}{P_{XY}[x, y]}\right)^2 - \left( \sum_{(x, y) \in \mathcal{D}} D_i[x, y] \left(1 + \log \frac{P_X[x]}{P_{XY}[x, y]}\right)\right)^2 \right) + \mathcal{O}(n_i^{-2})
\]

(1-\( \alpha \)) confidence interval: \( \left[ \max(0, pe - z_{\alpha/2} \sqrt{v}), \ pe + z_{\alpha/2} \sqrt{v} \right] \)
Quality vs cost of analysis

• Can optimize the sample size (the number of traces) for each component:

\[
\text{Estimated value of mutual information} = \frac{b_1}{n_1} + \frac{b_2}{n_2} + \frac{v_1}{n_1} + \frac{v_2}{n_2}
\]

Confidence interval

Adaptively updating the sample sizes \(n_i\) as

\[
\sqrt{\frac{v_i}{n}} \sum_{j=1}^{n} \sqrt{\frac{1}{v_j}}
\]
Abstraction-then-sampling

• Statistical method can be combined with qualitative program analysis:

```

<table>
<thead>
<tr>
<th>secret</th>
<th>obs =0</th>
<th>obs=1</th>
<th>obs=2</th>
<th>obs=3</th>
</tr>
</thead>
<tbody>
<tr>
<td>sec = 0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>sec = 1</td>
<td>0.025</td>
<td>0.050</td>
<td>0.075</td>
<td>0.100</td>
</tr>
<tr>
<td>sec = 2</td>
<td>0.025</td>
<td>0.050</td>
<td>0.075</td>
<td>0.100</td>
</tr>
<tr>
<td>sec = 3</td>
<td>0.025</td>
<td>0.050</td>
<td>0.075</td>
<td>0.100</td>
</tr>
<tr>
<td>sec = 4</td>
<td>0.025</td>
<td>0.050</td>
<td>0.075</td>
<td>0.100</td>
</tr>
<tr>
<td>sec = 5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
```

If we know these rows are identical … it’s sufficient to sample only one row.

By program analysis, we can find whether the output is independent of the secret inside a subsystem.

• More generally, prior knowledge on the system can reduce the cost of sampling.
Design of analysis tool

- Decompose a source code into components and choose an analysis type for each component:

  ![Diagram](image)

  - **Program analysis**
    - true: large matrix
    - false: many traces
  - **Statistical analysis**
    - true: many traces
    - false: large matrix

  - **Conditional**
    - true
    - false: Abstraction if possible

  - **Sample sizes**
    - True: $n_1$
    - False: $n_2$

  - **Approximate sub-distribution**
    - $b_1 / n_1$
    - $v_1 / n_1$

  - **Precise sub-distribution**
    - $b_2 / n_2$
    - $v_2 / n_2$

  - **Estimated distribution**
    - Estimated value of mutual information
    - Confidence interval

We apply a rough static heuristics to estimate the number of traces & the size of the matrix for each component.
Overview

• Model of information leakage
• Statistical method for estimating leakage amount
• Hybrid statistical method
• Evaluation of the method
• Summary & future work
Quality of the hybrid method

Larger samples size, then narrower confidence interval

More program analysis, then narrower confidence interval

When the sample size is $k$ times larger then the confidence interval is $\sqrt{k}$ times narrower.
Comparison of approaches

- Our hybrid method outperforms the **precise program analysis** and the **fully statistical analysis**:

<table>
<thead>
<tr>
<th></th>
<th>Reservoir</th>
<th>Lying Crypt</th>
<th>Window</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>N=6  N=8  N=10  N=12</td>
<td></td>
<td>N=20  N=22  N=24</td>
</tr>
<tr>
<td>Precise</td>
<td>Time(s)</td>
<td>0.7  11.4  timeout  timeout</td>
<td>506.4</td>
</tr>
<tr>
<td></td>
<td>Error</td>
<td>0     0     -       -</td>
<td>0</td>
</tr>
<tr>
<td>Statistical</td>
<td>Time(s)</td>
<td>21.6  35.2  60.7    91.5</td>
<td>254.3</td>
</tr>
<tr>
<td></td>
<td>Error</td>
<td>$10^{-3}$  $10^{-3}$  -       -</td>
<td>$10^{-3}$</td>
</tr>
<tr>
<td>Hybrid</td>
<td>Time(s)</td>
<td>13.4  22.5  34.6    58.4</td>
<td>240.1</td>
</tr>
<tr>
<td></td>
<td>Error</td>
<td>$10^{-4}$  $10^{-3}$  -       -</td>
<td>$10^{-3}$</td>
</tr>
</tbody>
</table>

*Table 2: Shannon leakage benchmark results.*

We are still trying to add more techniques from **formal methods**.
Overview

• Model of information leakage
• Statistical method for estimating leakage amount
• Hybrid statistical method
• Evaluation of the method
• Summary & future work
Summary & future work

• Showed a **hybrid method** for estimating information leakage that combines **statistical method** with **program analysis**.
  • Gets the best of the both methods to improve the quality & cost.

• Our ongoing work:
  Developing a tool for estimating information leakage by hybrid method.

• My future work:
  More fusion of **statistical methods & formal methods**.
Question/comments ?
References


