Hybrid Statistical Estimation of Mutual Information for Quantifying Information Flow

Yusuke Kawamoto, Fabrizio Biondi, Axel Legay

Hybrid Statistical Estimation of Mutual Information for Quantifying Information Flow ⋆

Yusuke Kawamoto[1], Fabrizio Biondi[2], and Axel Legay[2]

[1] AIST, Japan
[2] INRIA, France

Abstract. Analysis of a probabilistic system often requires to learn the joint probability distribution of its random variables. The computation of the exact distribution is usually an exhaustive *precise analysis* on all executions of the system. To avoid the high computational cost of such an exhaustive search, *statistical analysis* has been studied to efficiently obtain approximate estimates by analyzing only a small but representative subset of the system's behavior. In this paper we propose a *hybrid statistical estimation method* that combines precise and statistical analyses to estimate mutual information and its confidence interval. We show how to combine the analyses on different components of the system with different precision to obtain an estimate for the whole system. The new method performs weighted statistical analysis with different sample sizes over different components and dynamically finds their optimal sample sizes. Moreover it can reduce sample sizes by using prior knowledge about systems and a new *abstraction-then-sampling* technique based on qualitative analysis. We show the new method outperforms the state of the art in quantifying information leakage.

1 Introduction

In modeling and analyzing software and hardware systems, the statistical approach is often useful to evaluate quantitative aspects of the behaviors of the systems. In particular, probabilistic systems with complicated internal structures can be approximately and efficiently modeled and analyzed. For instance, statistical model checking has widely been used to verify quantitative properties of many kinds of probabilistic systems [40].

The *statistical analysis* of a probabilistic system is usually considered as a black-box testing approach in which the analyst does not require prior knowledge of the internal structure of the system. The analyst runs the system many times and records the execution traces to construct an approximate model of the system. Even when the formal specification or precise model of the system is not provided to the analyst, statistical analysis can be directly applied to the system if the analyst can execute the black-box implementation. Due to this random sampling of the systems, statistical analysis provides only approximate estimates. However, it can evaluate the accuracy and error of the analysis for instance by providing the confidence intervals of the estimated values.

One of the important challenges in statistical analysis is to estimate entropy-based properties in probabilistic systems. For example, statistical methods [13,19,21,20,8] have

⋆ This work was supported by JSPS KAKENHI Grant Number JP15H06886, by the MSR-Inria Joint Research Center, by the Sensation European grant, and by région Bretagne.

been studied for *quantitative information flow analysis* [22,38,14], which estimates an entropy-based property to quantify the leakage of confidential information in a system. More specifically, the analysis estimates *mutual information* or other properties between two random variables on the secrets and on the observable outputs in the system to measure the amount of information that is inferable about the secret by observing the output. The main technical difficulties in the estimation of entropy-based properties are

1. to efficiently compute large matrices that represent probability distributions, and
2. to provide a statistical method for correcting the bias of the estimate and computing a confidence interval to evaluate the accuracy of the estimation.

To overcome these difficulties we propose a method for statistically estimating mutual information, one of the most popular entropy-based properties. The new method, called *hybrid statistical estimation method*, integrates black-box statistical analysis and white-box *precise analysis*, exploiting the advantages of both. More specifically, this method employs some prior knowledge on the system and performs precise analysis (e.g., static analysis of the source code or specification) on some components of the system. Since precise analysis computes the exact sub-probability distributions of the components, the hybrid method using precise analysis is more accurate than statistical analysis alone.

Moreover, the new method can combine multiple statistical analyses on different components of the system to improve the accuracy and efficiency of the estimation. This is based on our new theoretical results that extend and generalize previous work [43,9,13] on purely statistical estimation. As far as we know this is the first work on a hybrid method for estimating entropy-based properties and their confidence intervals.

To illustrate the method we propose, Fig. 1 presents an example of a joint probability distribution P_{XY} between two random variables X and Y, built up from 3 overlapping components S_1, S_2 and T. To estimate the full joint distribution P_{XY}, the analyst separately computes the joint sub-distribution for the component T by precise analysis, estimates those for S_1 and S_2 by statistical analysis, and then combines these sub-distributions. Since the statistical analysis is based on the random sampling of execution traces, the empirical sub-distributions for S_1 and S_2 are different from the true ones, while the sub-distribution for T is exact. From these approximate and precise sub-distributions, the proposed method can estimate the mutual information for the entire system and evaluate its accuracy by providing a confidence interval. Owing to the combination of different kinds of analyses (with possibly different parameters such as sample sizes), the computation of the bias and confidence interval of the estimate is more complicated than the previous work on statistical analysis.

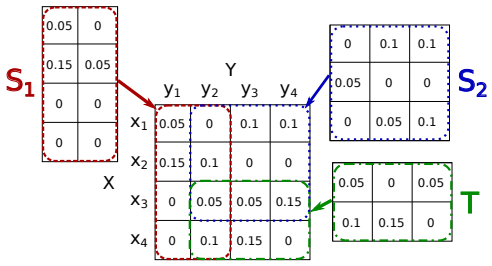


Fig. 1: Joint distribution composed of 3 components.

1.1 Contributions

The contributions of this paper are as follows:

- We propose a new method, called hybrid statistical estimation, that combines statistical and precise analyses on the estimation of mutual information (which can also be applied to Shannon entropy and conditional Shannon entropy). Specifically, we show theoretical results on compositionally computing the bias and confidence interval of the estimate from multiple statistical and precise analyses;
- We present a weighted statistical analysis method with different sample sizes over different components and a method for adaptively optimizing sample sizes for different components by evaluating the quality and cost of the analysis;
- We show how to reduce the sample sizes by using prior knowledge about systems, including an abstraction-then-sampling technique based on qualitative analysis;
- We show that the proposed method can be applied not only to composed systems but also to the source codes of a single system by decomposing it into components and determine the analysis method for each component;
- We evaluate the quality of the estimation in this method, showing that the estimates are more accurate than statistical analysis alone for the same sample size, and that the new method outperforms the state-of-the-art statistical analysis tool LeakWatch [20];
- We demonstrate the effectiveness of the hybrid method in case studies on the quantification of information leakage.

The rest of the paper is structured as follows. Section 2 introduces background in information theory and quantification of information. We compare precise analysis with statistical analysis for the estimation of mutual information. Section 3 describes the main results of this paper: the hybrid method for mutual information estimation, including the method for optimizing sample sizes for different components. Section 4 presents how to reduce sample sizes by using prior knowledge about systems, including the abstraction-then-sampling technique with qualitative analysis. Section 5 overviews how to decompose the source code of a system into components and to determine the analysis method for each component. Section 6 evaluates the proposed method and illustrates its effectiveness against the state of the art. Section 7 discusses related work and Section 8 concludes the paper. All proofs can be found in full version [35] of the paper.

2 Information Theory and Quantification of Information

In this section we introduce some background on information theory, which we use to quantify the amount of information in a system. We write X and Y to denote two random variables, and $\mathcal{X}$ and $\mathcal{Y}$ to denote the sets of all possible values of X and Y, respectively. We denote the number of elements of a set $\mathcal{S}$ by $\#\mathcal{S}$.

2.1 Channels

In information theory, a *channel* models the input-output relation of a system as a conditional probability distribution of outputs given inputs. This model has also been

used to formalize information leakage in a system that processes confidential data: *inputs* and *outputs* of a channel are respectively regarded as *secrets* and *observables* in the system and the channel represents relationships between the secrets and observables.

A *discrete channel* is a triple $(\mathcal{X}, \mathcal{Y}, C)$ where $\mathcal{X}$ and $\mathcal{Y}$ are two finite sets of discrete input and output values respectively and C is an $\#\mathcal{X} \times \#\mathcal{Y}$ matrix where each element $C[x, y]$ represents the conditional probability of an output y given an input x; i.e., for each $x \in \mathcal{X}$, $\sum_{y \in \mathcal{Y}} C[x, y] = 1$ and $0 \leq C[x, y] \leq 1$ for all $y \in \mathcal{Y}$.

A *prior* is a probability distribution on input values $\mathcal{X}$. Given a prior P_X over $\mathcal{X}$ and a channel C from $\mathcal{X}$ to $\mathcal{Y}$, the *joint probability distribution* P_{XY} of X and Y is defined by: $P_{XY}[x, y] = P_X[x]C[x, y]$ for each $x \in \mathcal{X}$ and $y \in \mathcal{Y}$.

2.2 Mutual Information

The amount of information gained about a random variable X by knowing a random variable Y is defined as the difference between the uncertainty about X before and after observing Y. The *mutual information* $I(X; Y)$ between X and Y is one of the most popular measures to quantify the amount of information on X gained/leaked by Y:

$$I(X; Y) = \sum_{x \in \mathcal{X}, y \in \mathcal{Y}} P_{XY}[x, y] \log_2 \left(\frac{P_{XY}[x, y]}{P_X[x] P_Y[y]} \right)$$

where P_Y is the marginal probability distribution defined as $P_Y[y] = \sum_{x \in \mathcal{X}} P_{XY}[x, y]$.

In the security scenario, information-theoretical measures quantify the amount of secret information leaked against some particular attacker: the mutual information between two random variables X on the secrets and Y on the observables in a system measures the information that is inferable about the secret by knowing the observable. In this scenario mutual information, or Shannon leakage, assumes an attacker that can ask binary questions on the secret's value after observing the system while min-entropy leakage [47] considers an attacker that has only one attempt to guess the secret's value.

Mutual information has been employed in many other applications including Bayesian networks [33], telecommunications [31], pattern recognition [28], machine learning [41], quantum physics [49], and biology [1]. In this work we focus on mutual information for the above security scenario.

2.3 Precise Analysis vs. Statistical Analysis

The calculation of the mutual information $I(X; Y)$ between input X and output Y in a probabilistic system requires the computation of the joint probability distribution P_{XY} of X and Y. The joint distribution can be computed precisely or estimated statistically.

Precise Analysis To obtain the exact joint probability $P_{XY}[x, y]$ for each $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, we sum the probabilities of all execution traces of the system that have input x and output y. This means the computation time depends on the number of traces in the system. If the system has a very large number of traces, it is intractable for analysts to precisely compute the joint distribution and consequently the mutual information.

In [50] the calculation of mutual information is shown to be computationally expensive. This computational difficulty comes from the fact that entropy-based properties are hyperproperties [25] that are defined using all execution traces of the system and therefore cannot be verified on each single trace. For example, when we investigate the leakage of confidential information in a system, it is insufficient to check the information leakage separately for each component of the system, because the attacker may derive sensitive information by combining the outputs of different components. More generally, the computation of entropy-based properties (such as the amount of leaked information) is not compositional in the sense that an entropy-based property of a system is not the (weighted) sum of those of the components.

For this reason it is inherently difficult to naïvely combine analyses of different components of a system to compute entropy-based properties. In fact, previous studies on the compositional approach in quantitative information flow analysis have faced certain difficulties in obtaining useful bounds on information leakage [4,29,36,37].

Statistical Analysis Due to the complexity of precise analysis, some previous studies have focused on computing approximate values of entropy-based measures. One of the common approaches is the *statistical analysis* based on Monte Carlo methods, in which approximate values are computed from repeated random sampling. Previous work on quantitative information flow has used statistical analysis to mutual information [13,43,9], channel capacity [13,8] and min-entropy leakage [20,16].

In the statistical estimation of mutual information between two random variables X and Y in a probabilistic system, analysts execute the system many times and collect the execution traces each recording a pair of values $(x, y) \in \mathcal{X} \times \mathcal{Y}$. This set of execution traces is used to estimate the empirical joint distribution $\hat{P}_{XY}$ of X and Y and then to compute the mutual information $I(X;Y)$.

Note that the empirical distribution $\hat{P}_{XY}$ is different from the true distribution P_{XY} and thus the estimated mutual information is different from the true value. In fact, it is known that entropy-based measures such as mutual information and min-entropy leakage have some bias and error that depends on the number of collected traces, the matrix size and other factors. However, results on statistics allow us to correct the bias of the estimate and to compute its 95% confidence interval. This way we can guarantee the quality of the estimation, which differentiates our approach from *testing*.

Comparing the Two Analysis Methods The cost of the statistical analysis is proportional to the size $\#\mathcal{X} \times \#\mathcal{Y}$ of the joint distribution matrix (strictly speaking, to the number of non-zero elements in the matrix). Therefore, this method is significantly more efficient than precise analysis if the matrix is relatively small and the number of all traces is very large (for instance because the system's internal variables have a large range). On the other hand, if the matrix is

	Precise	**Statistical**
Type	White box	Black/gray box
Analyzes	Source code	Implementation
Impractical for	Large number of traces	Large matrices
Produces	Exact value	Estimate & confidence

Table 1: Comparison of the two analysis methods.

very large, the number of executions needs to be very large to obtain a reliable and small confidence interval. In particular, for a small sample size, statistical analysis does not detect rare events, i.e., traces with a low probability that affect the result.

Main differences between precise and statistical analysis are summarized in Table 1.

3 Hybrid Statistical Estimation of Mutual Information

To overcome the above limitations on the previous approaches we introduce a new method, called *hybrid statistical estimation method*, that integrates both precise and statistical analyses. In this section we present the method for estimating the mutual information between two random variables X (over the inputs $\mathcal{X}$) and Y (over the outputs $\mathcal{Y}$) in a probabilistic system S, and for providing a confidence interval of this estimate. In the method we perform different types of analysis (with different parameters) on different components of a system.

- If a component is deterministic, we perform a precise analysis on it.
- If a component S_i has a joint sub-distribution matrix over *small* subsets of $\mathcal{X}$ and $\mathcal{Y}$ (relatively to the number of all traces), then we perform a statistical analysis on S_i.
- If a component T_j has a *large* matrix (relatively to the number of all traces), we perform a precise analysis on T_j.
- By combining the analysis results on all components we compute the mutual information estimate and its confidence interval. See the rest of Section 3 for details.
- By *qualitative* information flow analysis, the analyst may obtain partial knowledge on components and reduce the sample sizes. See Section 4 for details.

One of the main advantages of the new method is that we guarantee the quality of the outcome by providing its confidence interval even though different kinds of analyses with different parameters are combined together, such as multiple statistical analyses with different sample sizes.

Another advantage is the compositionality in estimating bias and confidence intervals. The random sampling of execution traces is performed independently for each component. Thanks to this we obtain that the bias and confidence interval of mutual information can be computed in a compositional way. This compositionality enables us to find optimal sample sizes for the different components that maximize the accuracy of the estimation (i.e., minimize the confidence interval size) given a fixed total sample size for the entire system. On the other hand, the computation of mutual information itself is not compositional; It requires calculating the *full* joint probability distribution of the system by summing the joint sub-distributions of all components of the system.

Note that these results can be applied to the estimation of Shannon entropy and conditional Shannon entropy as special cases. See the full version for the details.

3.1 Computation of Probability Distributions

We consider a probabilistic system S that consists of $(m+k)$ components $S_1, S_2, \ldots, S_m$ and $T_1, T_2, \ldots, T_k$ each executed with probabilities $\theta_1, \theta_2, \ldots, \theta_m$ and $\xi_1, \xi_2, \ldots, \xi_k$; i.e., when S is executed, it yields S_i with the probability θ_i and T_j with the probability

ξ_j. We assume S does not have non-deterministic transitions. Let $\mathcal{I} = \{1, 2, \ldots, m\}$ and $\mathcal{J} = \{1, 2, \ldots, k\}$, one of which can be empty. We assume the analyst can run the component S_i for each $i \in \mathcal{I}$ to record its execution traces, and precisely analyze the components T_j for $j \in \mathcal{J}$, e.g., by static analysis of the source code or specification.

In the estimation of mutual information between two random variables X and Y in the system S, we need to estimate the joint distribution P_{XY} of X and Y. In our approach this is obtained by combining the joint *sub-probability distributions* of X and Y for all the components S_i's and T_j's. More specifically, let R_i and Q_j be the joint sub-distributions of X and Y for the components S_i's and T_j's respectively. Then the joint (full) distribution P_{XY} for the whole system S is defined by:

$$P_{XY}[x, y] \stackrel{\text{def}}{=} \sum_{i \in \mathcal{I}} R_i[x, y] + \sum_{j \in \mathcal{J}} Q_j[x, y]$$

for $x \in \mathcal{X}$ and $y \in \mathcal{Y}$. Note that for each $i \in \mathcal{I}$ and $j \in \mathcal{J}$, the sums of all probabilities in R_i and Q_j equal the probabilities θ_i and ξ_j of executing S_i and T_j respectively.

To estimate the joint distribution P_{XY} the analyst computes

- for each $i \in \mathcal{I}$, the *empirical* sub-distribution $\hat{R}_i$ for the component S_i from a set of traces obtained by executing S_i, and
- for each $j \in \mathcal{J}$, the *exact* sub-distribution Q_j for T_j by a precise analysis on T_j.

The empirical sub-distribution $\hat{R}_i$ is constructed as follows. Let n_i be the number of S_i's executions. For each $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, let K_{ixy} be the number of S_i's traces that have input x and output y. Then $n_i = \sum_{x \in \mathcal{X}, y \in \mathcal{Y}} K_{ixy}$. From these we compute the empirical joint (full) distribution $\hat{D}_i$ of X and Y by $\hat{D}_i[x, y] \stackrel{\text{def}}{=} \frac{K_{ixy}}{n_i}$. Since S_i is executed with probability θ_i, $\hat{R}_i$ is given by $\hat{R}_i[x, y] \stackrel{\text{def}}{=} \theta_i \hat{D}_i[x, y] = \frac{\theta_i K_{ixy}}{n_i}$.

3.2 Estimation of Mutual Information and its Confidence Interval

In this section we present our new method for estimating mutual information and its confidence interval. For each component S_i let D_i be the joint (full) distribution of X and Y obtained by normalizing R_i: $D_i[x, y] = \frac{R_i[x,y]}{\theta_i}$. Let $D_{Xi}[x] = \sum_{y \in \mathcal{Y}} D_i[x, y]$, $D_{Yi}[y] = \sum_{x \in \mathcal{X}} D_i[x, y]$ and $\mathcal{D} = \{(x, y) \in \mathcal{X} \times \mathcal{Y} : P_{XY}[x, y] \neq 0\}$.

Using the estimated $\hat{P}_{XY}$ we can compute the mutual information estimate $\hat{I}(X; Y)$. Note that the mutual information of the whole system is smaller than (or equals) the weighted sum of those of the components, because of its convexity w.r.t. the channel matrix. Therefore it cannot be computed compositionally from those of the components; i.e., it requires to compute the joint distribution matrix $\hat{P}_{XY}$ for the whole system.

Since $\hat{I}(X; Y)$ is obtained from a limited number of traces, it is different from the true value $I(X; Y)$. The following theorem quantifies the bias $E(\hat{I}(X; Y)) - I(X; Y)$.

Theorem 1. *The expectation $E(\hat{I}(X; Y))$ of the mutual information is given by:*

$$I(X; Y) + \sum_{i \in \mathcal{I}} \frac{\theta_i^2}{2n_i} \Bigg(\sum_{(x,y) \in \mathcal{D}} \varphi_{ixy} - \sum_{x \in \mathcal{X}^+} \varphi_{ix} - \sum_{y \in \mathcal{Y}^+} \varphi_{iy} \Bigg) + \mathcal{O}(n_i^{-2})$$

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

The proof is based on the Taylor expansion w.r.t. multiple dependent variables and can be found in the full version. Since the higher-order terms in the formula are negligible when the sample sizes n_i are large enough, we use the following as the *point estimate*:

$$pe = \hat{I}(X;Y) - \sum_{i\in\mathcal{I}} \tfrac{\theta_i^2}{2n_i}\Big(\sum_{(x,y)\in\mathcal{D}} \widehat{\varphi}_{ixy} - \sum_{x\in\mathcal{X}^+} \widehat{\varphi}_{ix} - \sum_{y\in\mathcal{Y}^+} \widehat{\varphi}_{iy}\Big)$$

where $\widehat{\varphi}_{ixy}$, $\widehat{\varphi}_{ix}$ and $\widehat{\varphi}_{iy}$ are empirical values of φ_{ixy}, φ_{ix} and φ_{iy} respectively (that are computed from traces). Then the bias is closer to 0 when the sample sizes n_i are larger.

The quality of the estimate depends on the sample sizes n_i and other factors. The sampling distribution of the estimate $\hat{I}(X;Y)$ tends to follow the normal distribution when n_i's are large enough. The following gives the variance of the distribution.

Theorem 2. *The variance $V(\hat{I}(X;Y))$ of the mutual information is given by*

$$\sum_{i\in\mathcal{I}} \tfrac{\theta_i^2}{n_i}\left(\sum_{(x,y)\in\mathcal{D}} D_i[x,y]\Big(1+\log\tfrac{P_X[x]P_Y[y]}{P_{XY}[x,y]}\Big)^2 - \Big(\sum_{(x,y)\in\mathcal{D}} D_i[x,y]\Big(1+\log\tfrac{P_X[x]P_Y[y]}{P_{XY}[x,y]}\Big)\Big)^2\right) + \mathcal{O}(n_i^{-2})$$

The confidence interval of the estimate of mutual information is useful to know how accurate the estimate is. When the interval is smaller, we learn the estimate is more accurate. The confidence interval is calculated using the variance v obtained by Theorem 2. Given a significance level α, we denote by $z_{\alpha/2}$ the *z-score* for the $100(1-\frac{\alpha}{2})$ percentile point. Then *the* $(1-\alpha)$ *confidence interval* of the estimate is given by:

$$[\,\max(0, pe - z_{\alpha/2}\sqrt{v}),\ \ pe + z_{\alpha/2}\sqrt{v}\,]\ .$$

For example, we use the z-score $z_{0.0025} = 1.96$ to compute the 95% confidence interval. To ignore the higher order terms the sample size $\sum_{i\in\mathcal{I}} n_i$ needs to be at least $4{\cdot}\#\mathcal{X}{\cdot}\#\mathcal{Y}$.

By Theorems 1 and 3, the bias and confidence interval for the whole system can be computed compositionally from those for the components, unlike the mutual information itself. This allows us to adaptively optimize the sample sizes for the components.

3.3 Adaptive Optimization of Sample Sizes

The computational cost of the statistical analysis of each component S_i generally depends on the sample size n_i and the cost of each execution of S_i. When we choose n_i we take into account the trade-off between quality and cost of the analysis: a larger sample size provides a smaller confidence interval, while the cost increases proportionally to n_i.

In this section we present a method for deciding how many times we should run each component S_i to collect a sufficient number of traces to estimate mutual information. More specifically, we show how to compute optimal sample sizes n_i that achieves the smallest confidence interval size within the budget of the total sample size $n = \sum_{i\in\mathcal{I}} n_i$.

To compute the optimal sample sizes, we first run each component to collect a smaller number (for instance dozens) of execution traces. Then we calculate certain intermediate values in computing the variance to determine sample sizes for further executions. Formally, let v_i be the following intermediate value of the variance for S_i:

$$v_i = \theta_i^2\left(\sum_{(x,y)\in\mathcal{D}} \hat{D}_i[x,y]\Big(1+\log\tfrac{\hat{P}_X[x]\hat{P}_Y[y]}{\hat{P}_{XY}[x,y]}\Big)^2 - \Big(\sum_{(x,y)\in\mathcal{D}} \hat{D}_i[x,y]\Big(1+\log\tfrac{\hat{P}_X[x]\hat{P}_Y[y]}{\hat{P}_{XY}[x,y]}\Big)\Big)^2\right)$$

Then we find n_i's that minimize the variance $v = \sum_{i\in\mathcal{I}} \frac{v_i}{n_i}$ of the mutual information.

Theorem 3. *Given the total sample size n and the above intermediate variance v_i of the component S_i for each $i \in \mathcal{I}$, the variance of the mutual information estimate is minimized if, for all $i \in \mathcal{I}$, the sample size n_i for S_i satisfies $n_i = \frac{\sqrt{v_i}n}{\sum_{j=1}^{m}\sqrt{v_j}}$.*

By this result the estimation of a confidence interval size is useful to optimally assign sample sizes to components even when the analyst is not interested in the interval itself. We show experimentally the effectiveness of this optimization in the full version.

4 Estimation Using Prior Knowledge about Systems

In this section we show how to use prior knowledge about systems to improve the estimation, i.e., to make the size of the confidence intervals smaller and reduce the required sample sizes.

4.1 Approximate Estimation Using Knowledge of Prior Distributions

Our hybrid statistical estimation method integrates both precise and statistical analysis, and it can be seen as a generalization and extension of previous work [13,43,9].

For example, Chatzikokolakis et.al. [13] present a method for estimating mutual information between two random variables X (over secret values $\mathcal{X}$) and Y (over observable values $\mathcal{Y}$) when the analyst knows the (prior) distribution P_X of X. In the estimation they collect execution traces by running a system for each secret value $x \in \mathcal{X}$. Thanks to the precise knowledge of P_X, they have more accurate estimates than the other previous work [43,9] that also estimates P_X from execution traces.

Estimation using the precise knowledge of P_X is an instance of our result if a system is partitioned into the component S_x for each secret $x \in \mathcal{X} = \mathcal{I}$. If we assume all joint probabilities are non-zero, the approximate result in [13] follows from Theorem 1.

Corollary 1. *The expectation $E(\hat{I}(X;Y))$ of the mutual information is given by*

$$I(X;Y) + \frac{(\#\mathcal{X}-1)(\#\mathcal{Y}-1)}{2n} + \mathcal{O}(n^{-2}).$$

In this result from [13] the bias $\frac{(\#\mathcal{X}-1)(\#\mathcal{Y}-1)}{2n}$ depends only on the size of the joint distribution matrix. However, the bias can be strongly influenced by zeroes or very small probabilities in the distribution, therefore their approximate results can be correct only when all joint probabilities are non-zero and large enough, which is a strong restriction in practice. The tool LeakWatch [20] implicitly assumes that all probabilities are large enough, and consequently miscalculates bias and gives an estimate far from the true value in the presence of very small probabilities.

4.2 Our Estimation Using Knowledge of Prior Distributions

To overcome these issues we present more general results in the case the analyst knows the prior distribution P_X. We assume that a system S is partitioned into the disjoint component S_{ix} for each index $i \in \mathcal{I}$ and secret $x \in \mathcal{X}$, and that each S_{ix} is executed with probability θ_{ix} in the system S. Let $\Theta = \{\theta_{ix} : i \in \mathcal{I},\ x \in \mathcal{X}\}$.

In the estimation of mutual information we run each component S_{ix} separately many times to collect execution traces. Unlike the previous work we may change the number of executions $n_i P_X[x]$ to $n_i \lambda_i[x]$ where $\lambda_i[x]$ is an *importance prior* that decides how the sample size n_i is allocated for each component S_{ix}. Let $\Lambda = \{\lambda_i : i \in \mathcal{I}\}$.

Given the number K_{ixy} of S_{ix}'s traces with output y, we define the conditional distribution D_i of output given input: $D_i[y|x] \stackrel{\text{def}}{=} \frac{K_{ixy}}{n_i \lambda_i[x]}$. Let $M_{ixy} = \frac{\theta_{ix}^2}{\lambda_i[x]} D_i[y|x]\,(1 - D_i[y|x])$. Then the following is the expectation and variance of the mutual information $\hat{I}_{\Theta,\Lambda}(X;Y)$ calculated using $\hat{D}_i$, Θ, Λ.

Proposition 1. *The expectation $E(\hat{I}_{\Theta,\Lambda}(X;Y))$ of the mutual information is given by*

$$I(X;Y) + \sum_{i \in \mathcal{I}} \frac{1}{2n_i} \sum_{y \in \mathcal{Y}^+} \left(\sum_{x \in \mathcal{D}_y} \frac{M_{ixy}}{P_{XY}[x,y]} - \frac{\sum_{x \in \mathcal{D}_y} M_{ixy}}{P_Y[y]} \right) + \mathcal{O}(n_i^{-2})$$

Proposition 2. *The variance $V(\hat{I}_{\Theta,\Lambda}(X;Y))$ of the mutual information is given by*

$$\sum_{i \in \mathcal{I}} \sum_{x \in \mathcal{X}^+} \frac{\theta_{ix}^2}{n_i \lambda_i[x]} \left(\sum_{y \in \mathcal{D}_x} D_i[y|x] \left(\log \frac{P_Y[y]}{P_{XY}[x,y]} \right)^2 - \left(\sum_{y \in \mathcal{D}_x} D_i[y|x] \left(\log \frac{P_Y[y]}{P_{XY}[x,y]} \right) \right)^2 \right) + \mathcal{O}(n_i^{-2})$$

By applying Theorem 3, the sample sizes n_i and the importance priors λ_i can be adaptively optimized.

4.3 Abstraction-Then-Sampling Using Partial Knowledge of Components

In this section we extend our estimation method to consider the case in which the analyst has partial knowledge of components (e.g. by static analysis of the source code or specification) before sampling. Such prior knowledge may help us abstract components into simpler ones and thus reduce the sample size for the statistical analysis.

For instance, let us consider an analyst who knows two pairs (x, y) and (x', y') of inputs and outputs have the same probability in a component S_i: $D_i[x, y] = D_i[x', y']$. Then, when we construct the empirical distribution $\hat{D}_i$ from a set of traces, we can count the number $K_{i\{(x,y),(x',y')\}}$ of traces having either (x, y) or (x', y'), and divide it by two: $K_{ixy} = K_{ix'y'} = \frac{K_{i\{(x,y),(x',y')\}}}{2}$. Then the sample size required for a certain accuracy is smaller than when we do not use the prior knowledge on the equality $K_{ixy} = K_{ix'y'}$.

In the following we generalize this idea to deal with more knowledge of components. Let us consider a (probabilistic) system in which some components leak no information on inputs and the analyst can learn this by *qualitative* information analysis (for verifying non-interference). Then such a component S_i has a sub-channel matrix where all non-zero rows have an identical conditional distribution of outputs given inputs [26]. Consequently, when we estimate the $\#\mathcal{X}_i \times \#\mathcal{Y}_i$ matrix of S_i it suffices to estimate one of the rows, hence the number of executions is proportional to $\#\mathcal{Y}_i$ instead of $\#\mathcal{X}_i \times \#\mathcal{Y}_i$. Note that even when some components leak no information, computing the mutual information for the whole system requires constructing the matrix of the system, hence the matrices of all components.

The following results show that the bias and confidence interval are narrower than when not using the prior knowledge of components. Let $\mathcal{I}^\star$ be the set of indices of components that have channel matrices whose non-zero rows consist of the same distribution. For each $i \in \mathcal{I}^\star$, we define $\pi_i[x]$ as the probability of having an input x in the component S_i. Then the expectation and variance of the mutual information are as follows.

Theorem 4. *The expectation $E(\hat{I}_{\mathcal{I}^\star}(X;Y))$ of the mutual information is given by*

$$I(X;Y)+\sum_{i\in\mathcal{I}\setminus\mathcal{I}^\star}\frac{\theta_i^2}{2n_i}\Big(\sum_{(x,y)\in\mathcal{D}}\varphi_{ixy}-\sum_{x\in\mathcal{X}^+}\varphi_{ix}-\sum_{y\in\mathcal{Y}^+}\varphi_{iy}\Big)+\sum_{i\in\mathcal{I}^\star}\frac{\theta_i^2}{2n_i}\Big(\sum_{(x,y)\in\mathcal{D}}\psi_{ixy}-\sum_{y\in\mathcal{Y}^+}\varphi_{iy}\Big)+\mathcal{O}(n_i^{-2})$$

where $\psi_{ixy} \stackrel{\text{def}}{=} \frac{D_i[x,y]\pi_i[x]-D_i[x,y]^2}{P_{XY}[x,y]}$.

Theorem 5. *The variance $V(\hat{I}_{\mathcal{I}^\star}(X;Y))$ of the mutual information is given by*

$$\sum_{i\in\mathcal{I}\setminus\mathcal{I}^\star}\frac{\theta_i^2}{n_i}\Bigg(\sum_{(x,y)\in\mathcal{D}}D_i[x,y]\Big(1+\log\tfrac{P_X[x]P_Y[y]}{P_{XY}[x,y]}\Big)^2-\Big(\sum_{(x,y)\in\mathcal{D}}D_i[x,y]\Big(1+\log\tfrac{P_X[x]P_Y[y]}{P_{XY}[x,y]}\Big)\Big)^2\Bigg)$$
$$+\sum_{i\in\mathcal{I}^\star}\frac{\theta_i^2}{n_i}\Bigg(\sum_{y\in\mathcal{Y}^+}D_{Yi}[y]\Big(\log P_Y[y]-\sum_{x\in\mathcal{X}}\pi_i[x]\log P_{XY}[x,y]\Big)^2$$
$$-\Big(\sum_{y\in\mathcal{Y}^+}D_{Yi}[y]\Big(\log P_Y[y]-\sum_{x\in\mathcal{X}}\pi_i[x]\log P_{XY}[x,y]\Big)\Big)^2\Bigg)+\mathcal{O}(n_i^{-2})\,.$$

5 Estimation via Program Decomposition

The hybrid statistical estimation presented in the previous sections is designed to analyze a system composed of subsystems (for instance, a distributed system over different software or hardware, potentially geographically separated). However, it can also be applied to the the source code of a system by decomposing it into disjoint components. In this section we show how to decompose a code into components and determine for each component which analysis method to use and the method's parameters.

The principles to decompose a system's source code in components are as follows:

- The code may be decomposed only at conditional branching. Moreover, each component must be a terminal in the control flow graph, hence no component is executed afterwards. This is because the estimation method requires that the channel matrix for the system is the weighted sum of those for its components, and that the weight of a component is the probability of executing it.
- The analysis method and its parameters for each component S_i are decided by estimating the computational cost of analyzing S_i. Let $\mathcal{Z}_i$ be the set of all *internal randomness* (i.e., the variables whose values are assigned according to probability distributions) in S_i. Then the cost of the statistical analysis is proportional to S_i's matrix size $\#\mathcal{X}_i \times \#\mathcal{Y}_i$, while the cost of the precise analysis is proportional to the number of all traces in S_i's control flow graph (in the worst case proportional to $\#\mathcal{X}_i \times \#\mathcal{Z}_i$). Hence the cost estimation is reduced to counting $\#\mathcal{Y}_i$ and $\#\mathcal{Z}_i$.

The procedure for decomposition is shown in Fig. 2. Since this is heuristic, it is not guaranteed to produce an optimal decomposition. While the procedure is automated, for usability the choice of analysis can be controlled by user's annotations on the code.

1. Build the control flow graph of the system.
2. Mark all possible components based on each conditional branching. Each possible component must be a terminal as explained in Section 5.
3. For each possible component S_i, check whether it is deterministic or not (by syntactically checking an occurrence of a probabilistic assignment or a probabilistic function call). If it is, mark the component for precise analysis.
4. For each possible component S_i, check whether S_i's output variables are independent of its input variables inside S_i (by *qualitative* information flow). If so, mark that the abstraction-then-sampling technique in Section 4.3 is to be used on the component.
5. For each S_i, estimate an approximate range size of its internal and observable variables.
6. Looking from the leaves to the root of the graph, decide the decomposition into components. Estimate the cost of statistical and precise analyses and mark the component for analysis by the cheapest of the two.
7. Join together adjacent components if they are marked for precise analysis, or if they are marked for statistical analysis and have the same input and output ranges.
8. For each component, perform precise analysis or statistical analysis as marked.

Fig. 2: Procedure for decomposing a system given its source code.

6 Evaluation

We evaluate experimentally the effectiveness of our hybrid method compared to the state of the art. We first discuss the cost and quality of the estimation, then test the hybrid method against fully precise/fully statistical analyses on Shannon leakage benchmarks.

6.1 On the Tradeoff between the Cost and Quality of Estimation

In the hybrid statistical estimation, the estimate takes different values probabilistically, because it is computed from a set of traces that are generated by executing a probabilistic system. Fig. 3 shows the sampling distribution of the mutual information estimate of the joint distribution in Fig. 1 in Section 1. The graph shows the frequency (on the y axis) of the mutual information estimates (on the x axis) when performing the estimation 1000 times. In each estimation we perform precise analysis on the component T and statistical analysis on S_1 and S_2 (with a sample size of 5000). As shown in Fig. 3 the estimate after the correction of bias by Theorem 1 is closer to the true value. The estimate is roughly between the lower and upper bounds of the 95% confidence interval calculated using Theorem 2.

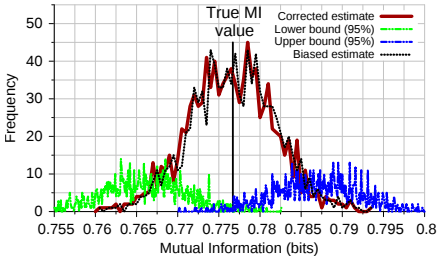
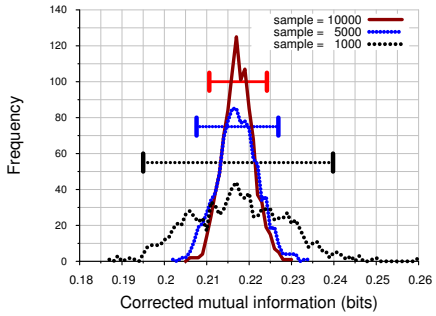
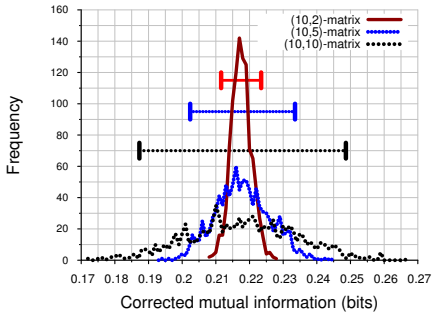


Fig. 3: Distribution of mutual information estimate and its confidence interval.



(a) Estimates and sample sizes.



(b) Estimates and the ratio of precise analysis.

Fig. 4: Smaller intervals when increasing the sample size or the ratio of precise analysis.

The interval size depends on the sample size in statistical analysis as shown in Fig. 4a. When the sample size is k times larger, the confidence interval is $\sqrt{k}$ times narrower. The interval size also depends on the amount of precise analysis as shown in Fig. 4b. If we perform precise analysis on larger components, then the sampling distribution becomes more centered (with shorter tails) and the confidence interval becomes narrower.

The hybrid approach produces better estimates than the state of the art in statistical analysis. Due to the combination with precise analysis, the confidence interval estimated by our approach is smaller than LeakWatch [20] for the same sample size.

6.2 Shannon Leakage Benchmarks

We compare the performance of our hybrid method with fully precise/statistical analysis on Shannon leakage benchmarks. Our implementations of precise and statistical analyses are variants of the state-of-the art tools QUAIL [7,6] and LeakWatch [20,17] respectively. They are fully automated except for human-provided annotations to determine the analysis method for each component. All experiments are performed on an Intel i7-4960HQ 2.6GHz quad-core machine with 8GB of RAM running Ubuntu 16.04 .

```
1 secret array bit[N] s;
2 observable array bit[K] r;
3 for i=0..K-1 do r[i]=s[i] ;
4 for i=K..N-1 do
5 |   j = uniform(0..i);
6 |   if j<K then r[j]=s[i];
7 end
```

Fig. 5: Reservoir sampling.

Reservoir Sampling The reservoir sampling problem [48] consists of selecting K elements randomly from a pool of $N > K$ elements. We quantify the information flow of the commonly-used *Algorithm R* [48], presented in Fig 5, for various values of N and $K = N/2$. In the algorithm, the first K elements are chosen as the sample, then each other element has a probability to replace one element in the sample.

```
1  secret int h = [0, N];
2  observable array bit[N] decl;
3  int lie = uniform(1..N);
4  randomly generated array bit[N] coin;
5  for c in coin do  c = uniform(0..1) ;
6  for i=0..N-1 do
7  |  decl[i]=coin[i] xor coin[(i+1)%N];
8  |  if h==i+1 then decl[i]=!decl[i];
9  |  if i==lie then decl[i]=!decl[i];
10 end
```

Fig. 6: Lying cryptographers.

Multiple Lying Cryptographers Protocol We test our hybrid method to compute the Shannon leakage of a distributed version of the lying cryptographers protocol. The lying cryptographers protocol is a variant of the dining cryptographer multiparty computation protocol [15] in which a randomly-chosen cryptographer declares the opposite of what they would normally declare, i.e. they lie if they are not the payer, and do not lie if they are the payer. We consider three simultaneous lying cryptographers implementation in which 8 cryptographers run the protocol on three separate overlapping tables A, B and C with 4 cryptographers each. Table A hosts cryptographers 1 to 4, Table B hosts cryptographers 3 to 6, and Table C hosts cryptographers 5 to 8. The identity of the payer is the same in all tables.

```
1  secret int sec = [0,N-1];
2  observable int obs;
3  int S = uniform(0,N-W-1);
4  int ws = uniform(1,W);
5  int O = uniform(0,N-W-1);
6  int wo = uniform(1,W);
7  if S ≤ sec ≤ S+ws then
8  |  obs = uniform(O,O+wo);
9  else
10 |  obs = uniform(0,N-1);
11 end
```

Fig. 7: Shifting Window.

Shifting Window In the shifting window example the secret has N possible values, and a contiguous sequence of this values (the "window") of random size from 1 to W is chosen. We assume for simplicity that $N = 2W$. If the secret is inside the window then another random window is chosen in the same way and a random value from the new window is printed. Otherwise, a random value from 0 to $N - 1$ is printed.

Results In Table 2 we show the results of the benchmarks using fully precise, fully statistical and hybrid analyses, for a sample size of 100000 executions. Timeout is set at 10 minutes. On the reservoir benchmark the precise analysis is faster for small instances but does not scale, timing out on larger values of N. The hybrid method is consistently faster than the fully statistical analy-

		Reservoir				Lying Crypt	Window		
		N=6	**N=8**	**N=10**	**N=12**		**N=20**	**N=22**	**N=24**
Precise	**Time(s)**	0.7	11.4	timeout	timeout	506.4	10.0	16.0	28.3
	Error	0	0	-	-	0	0	0	0
Statistical	**Time(s)**	21.6	35.2	60.7	91.5	254.3	7.5	7.7	7.1
	Error	10^{-3}	10^{-3}	-	-	10^{-3}	10^{-3}	10^{-3}	10^{-4}
Hybrid	**Time(s)**	13.4	22.5	34.6	58.4	240.1	6.6	7.1	7.1
	Error	10^{-4}	10^{-3}	-	-	10^{-3}	10^{-7}	10^{-4}	10^{-4}

Table 2: Shannon leakage benchmark results.

sis and often has a smaller error. On the other benchmarks the hybrid method usually outperforms the others and produces better approximations than the statistical analysis.

The results in Table 2 show the superiority of our hybrid approach compared to the state of the art. The hybrid analysis scales better than the precise analysis, since it does not need to analyze every trace of the system. Compared to fully statistical analysis, our hybrid analysis exploits precise analysis on components of the system where statistical estimation would be more expensive than precise analysis. This allows the hybrid analysis to focus the statistical estimation on components of the system where it converges faster, thus obtaining a smaller confidence interval in a shorter time.

7 Related Work

The information-theoretical approach to program security dates back to the work of Denning [27] and Gray [32]. Clark et al. [22,23] presented techniques to automatically compute mutual information of an imperative language with loops. For a deterministic program, leakage can be computed from the equivalence relations on the secret induced by the possible outputs, and such relations can be automatically quantified [2]. Under- and over-approximation of leakage based on the observation of some traces have been studied for deterministic programs [42,44]. The combination of static and statistical approaches to quantitative information flow is proposed in [39] while our paper is general enough to deal with probabilistic systems under various prior information conditions.

The statistical approach to quantifying information leakage has been studied since the seminal work by Chatzikokolakis et al. [13]. Chothia et al. have developed this approach in tools leakiEst [19,18] and LeakWatch [20,17]. The hybrid statistical method in this paper can be considered as their extension with the inclusion of component weighting and adaptive priors inspired by the importance sampling in statistical model checking [3,24]. To the best of our knowledge, no prior work has applied weighted statistical analysis to the estimation of mutual information or any other leakage measures.

Fremont and Seshia [30] have presented a polynomial time algorithm to approximate the weight of traces of deterministic programs with possible application to quantitative information leakage. Progress in statistical program analysis includes a scalable algorithm for uniform generation of sample from a distribution defined as constraints [11,12], with applications to constrained-random program verification.

The algorithms for precise computation of information leakage used in this paper are based on trace analysis [5], implemented in the QUAIL tool [6,7]. Phan et al. [45,46] developed tools to compute channel capacity of deterministic programs written in the C or Java languages. McCamant et al. [34] developed tools implementing dynamic quantitative taint analysis techniques for security. The recent tool Moped-QLeak [10] is able to efficiently compute information leakage of programs as long as it can produce a complete symbolic representation of the program.

8 Conclusions and Future Work

We have proposed a method for estimating mutual information by combining precise and statistical analyses and for compositionally computing the bias and confidence interval

of the estimate. The results are also used to adaptively find the optimal sample sizes for different components in the statistical analysis. Moreover, we have shown how to reduce sample sizes by using prior knowledge about systems, including the abstraction-then-sampling technique with qualitative analysis. To apply our new method to the source codes of systems we have shown how to decompose the codes into components and determine the analysis method for each component. We have shown both theoretical and experimental results to demonstrate that the proposed approach outperforms the state of the art. To obtain better results we are developing theory and tools that integrate symbolic abstraction techniques in program analysis into our estimation method.

References

1. Adami, C.: Information theory in molecular biology. Physics of Life Reviews 1(1), 3–22 (Apr 2004)
2. Backes, M., Köpf, B., Rybalchenko, A.: Automatic discovery and quantification of information leaks. In: 30th IEEE Symposium on Security and Privacy (S&P 2009), 17-20 May 2009, Oakland, California, USA. pp. 141–153. IEEE Computer Society (2009)
3. Barbot, B., Haddad, S., Picaronny, C.: Coupling and importance sampling for statistical model checking. In: Flanagan, C., König, B. (eds.) TACAS 2012. Proceedings. Lecture Notes in Computer Science, vol. 7214, pp. 331–346. Springer (2012)
4. Barthe, G., Köpf, B.: Information-theoretic bounds for differentially private mechanisms. In: Proc. of CSF. pp. 191–204. IEEE (2011)
5. Biondi, F., Legay, A., Malacaria, P., Wasowski, A.: Quantifying information leakage of randomized protocols. Theor. Comput. Sci. 597, 62–87 (2015)
6. Biondi, F., Legay, A., Traonouez, L.M., Wasowski, A.: QUAIL, https://project.inria.fr/quail/
7. Biondi, F., Legay, A., Traonouez, L., Wasowski, A.: QUAIL: A quantitative security analyzer for imperative code. In: Sharygina, N., Veith, H. (eds.) CAV 2013. Proceedings. Lecture Notes in Computer Science, vol. 8044, pp. 702–707. Springer (2013)
8. Boreale, M., Paolini, M.: On formally bounding information leakage by statistical estimation. In: Chow, S.S.M., Camenisch, J., Hui, L.C.K., Yiu, S. (eds.) ISC 2014. Proceedings. Lecture Notes in Computer Science, vol. 8783, pp. 216–236. Springer (2014)
9. Brillinger, D.R.: Some data analysis using mutual information. Brazilian Journal of Probability and Statistics 18(6), 163–183 (2004)
10. Chadha, R., Mathur, U., Schwoon, S.: Computing information flow using symbolic model-checking. In: Raman, V., Suresh, S.P. (eds.) FSTTCS 2014. Proceedings. LIPIcs, vol. 29, pp. 505–516. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2014)
11. Chakraborty, S., Fremont, D.J., Meel, K.S., Seshia, S.A., Vardi, M.Y.: On parallel scalable uniform SAT witness generation. In: Baier, C., Tinelli, C. (eds.) TACAS 2015. Proceedings. Lecture Notes in Computer Science, vol. 9035, pp. 304–319. Springer (2015)
12. Chakraborty, S., Meel, K.S., Vardi, M.Y.: A scalable approximate model counter. In: Schulte, C. (ed.) CP 2013. Proceedings. Lecture Notes in Computer Science, vol. 8124, pp. 200–216. Springer (2013)
13. Chatzikokolakis, K., Chothia, T., Guha, A.: Statistical measurement of information leakage. In: Esparza, J., Majumdar, R. (eds.) TACAS 2010. Proceedings. Lecture Notes in Computer Science, vol. 6015, pp. 390–404. Springer (2010)
14. Chatzikokolakis, K., Palamidessi, C., Panangaden, P.: Anonymity protocols as noisy channels. Inf. and Comp. 206(2–4), 378–401 (2008)
15. Chaum, D.: The dining cryptographers problem: Unconditional sender and recipient untraceability. Journal of Cryptology 1, 65–75 (1988)

16. Chothia, T., Kawamoto, Y.: Statistical estimation of min-entropy leakage (April 2014), http://www.cs.bham.ac.uk/research/projects/infotools/, manuscript
17. Chothia, T., Kawamoto, Y., Novakovic, C.: LeakWatch, http://www.cs.bham.ac.uk/research/projects/infotools/leakwatch/
18. Chothia, T., Kawamoto, Y., Novakovic, C.: leakiEst, http://www.cs.bham.ac.uk/research/projects/infotools/leakiest/
19. Chothia, T., Kawamoto, Y., Novakovic, C.: A tool for estimating information leakage. In: Sharygina, N., Veith, H. (eds.) CAV 2013. Proceedings. Lecture Notes in Computer Science, vol. 8044, pp. 690–695. Springer (2013)
20. Chothia, T., Kawamoto, Y., Novakovic, C.: Leakwatch: Estimating information leakage from java programs. In: Kutylowski, M., Vaidya, J. (eds.) ESORICS 2014. Proceedings, Part II. Lecture Notes in Computer Science, vol. 8713, pp. 219–236. Springer (2014)
21. Chothia, T., Kawamoto, Y., Novakovic, C., Parker, D.: Probabilistic point-to-point information leakage. In: CSF 2013. Proceedings. pp. 193–205. IEEE (2013)
22. Clark, D., Hunt, S., Malacaria, P.: Quantitative analysis of the leakage of confidential data. Electr. Notes Theor. Comput. Sci. 59(3), 238–251 (2001)
23. Clark, D., Hunt, S., Malacaria, P.: A static analysis for quantifying information flow in a simple imperative language. Journal of Computer Security 15(3), 321–371 (2007)
24. Clarke, E.M., Zuliani, P.: Statistical model checking for cyber-physical systems. In: Bultan, T., Hsiung, P. (eds.) ATVA 2011. Proceedings. Lecture Notes in Computer Science, vol. 6996, pp. 1–12. Springer (2011)
25. Clarkson, M.R., Schneider, F.B.: Hyperproperties. Journal of Computer Security 18(6), 1157–1210 (2010)
26. Cover, T.M., Thomas, J.A.: Elements of information theory (2. ed.). A Wiley-Interscience publication, Wiley (2006)
27. Denning, D.E.: A lattice model of secure information flow. Commun. ACM 19(5), 236–243 (1976)
28. Escolano, F., Suau, P., Bonev, B.: Information Theory in Computer Vision and Pattern Recognition. Springer, Londres (2009), http://opac.inria.fr/record=b1130015
29. Espinoza, B., Smith, G.: Min-entropy as a resource. Inf. Comput. (2013)
30. Fremont, D.J., Seshia, S.A.: Speeding up SMT-based quantitative program analysis. In: Rümmer, P., Wintersteiger, C.M. (eds.) SMT 2014. Proceedings. CEUR Workshop Proceedings, vol. 1163, pp. 3–13. CEUR-WS.org (2014)
31. Gallager, R.G.: Information Theory and Reliable Communication. John Wiley & Sons, Inc., New York, NY, USA (1968)
32. Gray, J.W.: Toward a mathematical foundation for information flow security. In: IEEE Symposium on Security and Privacy. pp. 21–35 (1991)
33. Jensen, F.V.: Introduction to Bayesian Networks. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1st edn. (1996)
34. Kang, M.G., McCamant, S., Poosankam, P., Song, D.: DTA++: dynamic taint analysis with targeted control-flow propagation. In: NDSS 2011. Proceedings. The Internet Society (2011)
35. Kawamoto, Y., Biondi, F., Legay, A.: Hybrid Statistical Estimation of Mutual Information for Quantifying Information Flow. Research report, INRIA (2016), available at https://hal.inria.fr/hal-01241360
36. Kawamoto, Y., Chatzikokolakis, K., Palamidessi, C.: Compositionality results for quantitative information flow. In: QEST 2014. Proceedings. pp. 368–383 (2014)
37. Kawamoto, Y., Given-Wilson, T.: Quantitative information flow for scheduler-dependent systems. In: QAPL 2015. Proceedings. vol. 194, pp. 48–62 (2015)
38. Köpf, B., Basin, D.A.: An information-theoretic model for adaptive side-channel attacks. In: Proc. of CCS. pp. 286–296. ACM (2007)

39. Köpf, B., Rybalchenko, A.: Approximation and randomization for quantitative information-flow analysis. In: CSF 2010. Proceedings. pp. 3–14. IEEE Computer Society (2010)
40. Legay, A., Delahaye, B., Bensalem, S.: Statistical model checking: An overview. In: Barringer, H., Falcone, Y., Finkbeiner, B., Havelund, K., Lee, I., Pace, G.J., Rosu, G., Sokolsky, O., Tillmann, N. (eds.) Runtime Verification - First International Conference, RV 2010, St. Julians, Malta, November 1-4, 2010. Proceedings. Lecture Notes in Computer Science, vol. 6418, pp. 122–135. Springer (2010)
41. MacKay, D.J.C.: Information Theory, Inference & Learning Algorithms. Cambridge University Press, New York, NY, USA (2002)
42. McCamant, S., Ernst, M.D.: Quantitative information flow as network flow capacity. In: Gupta, R., Amarasinghe, S.P. (eds.) Proceedings of the ACM SIGPLAN 2008 Conference on Programming Language Design and Implementation, Tucson, AZ, USA, June 7-13, 2008. pp. 193–205. ACM (2008)
43. Moddemeijer, R.: On estimation of entropy and mutual information of continuous distributions. Signal Processing 16, 233–248 (1989)
44. Newsome, J., McCamant, S., Song, D.: Measuring channel capacity to distinguish undue influence. In: Chong, S., Naumann, D.A. (eds.) Proceedings of the 2009 Workshop on Programming Languages and Analysis for Security, PLAS 2009, Dublin, Ireland, 15-21 June, 2009. pp. 73–85. ACM (2009)
45. Phan, Q., Malacaria, P.: Abstract model counting: a novel approach for quantification of information leaks. In: Moriai, S., Jaeger, T., Sakurai, K. (eds.) AsiaCCS 2014. Proceedings. pp. 283–292. ACM (2014)
46. Phan, Q., Malacaria, P., Pasareanu, C.S., d'Amorim, M.: Quantifying information leaks using reliability analysis. In: Rungta, N., Tkachuk, O. (eds.) SPIN 2014. Proceedings. pp. 105–108. ACM (2014)
47. Smith, G.: On the foundations of quantitative information flow. In: de Alfaro, L. (ed.) FOSSACS 2009. Proceedings. Lecture Notes in Computer Science, vol. 5504, pp. 288–302. Springer (2009)
48. Vitter, J.S.: Random sampling with a reservoir. ACM Trans. Math. Softw. 11(1), 37–57 (Mar 1985), http://doi.acm.org/10.1145/3147.3165
49. Wilde, M.M.: Quantum Information Theory. Cambridge University Press, New York, NY, USA, 1st edn. (2013)
50. Yasuoka, H., Terauchi, T.: Quantitative information flow as safety and liveness hyperproperties. Theor. Comput. Sci. 538, 167–182 (2014)