Quantitative Information Flow Metrics

Ji Zhu[†], Mudhakar Srivatsa[‡] and Bruce Hajek[†]
Dept of Electrical and Computer Engg, University of Illinois at Urbana-Champaign[†]
IBM T.J. Watson Research Center[‡]
jizhu1@illinois.edu, msrivats@us.ibm.com, b-hajek@uiuc.edu

I. Overview

Information flow analysis is a powerful technique for reasoning about sensitive information that may be exposed during program execution. One promising approach is to adopt a *program as a communication channel* model and leverage information theoretic metrics (e.g., mutual information between the sensitive input and the public output) to quantify such information flows. However, recent research has shown discrepancies in such information theoretic metrics: for example, Smith et. al. [5] showed examples wherein using the classical Shannon entropy measure for quantifying information flows may be counter-intuitive. Smith et. al. [5] proposed a vulnerability measure in an attempt to resolve this problem; this measure was subsequently enhanced by Hamadou et. al. [2] into a belief-vulnerability metric (in Oakland 2010). However, we point out that the vulnerability measure may also lead to counter-intuitive results on several other programs. In fact, we show that one can construct infinitely many programs wherein different information leakage measures (proposed in past work) are in conflict. This paper presents the first attempt towards addressing such conflicts and derives solutions for an *optimal conflict-free* comparison of programs over a class of entropy measures (called Renyi entropy − a well known generalization of the classical Shannon entropy).

II. Quantifying Information Leakage

Past work on quantitative information leakage metrics has explored using several entropy measures to compute mutual information, including, Shannon entropy, min-entropy, Guessing entropy (see [2, 3, 5] for more details), and so on. However, in most past work, the choice of such entropy measure has been ad hoc (mostly driven by sample programs) − often leading to counter-intuitive results. Consider the following two programs (by Smith[5]), where the secret input A is uniformly distributed $8k$-bit integer with $k \geq 2$, & denotes bitwise *and* operator and $0^{7k-1}1^{k+1}$ denotes a binary constant.

PROG P1
if $A \equiv 0 \mod 8$ **then**
$O = A$
else
$O = 1$
end if

and

PROG P2
$O = A \ \& \ 0^{7k-1}1^{k+1}$

Intuitively, one might argue that PROG P1 leaks more information leakage than PROG P2 when k is large, because it reveals complete information about the secret input with probability $\frac{1}{8}$; on the other hand, when k is large, PROG P2 reveals roughly $\frac{1}{8}$ of the number of bits in A. However, applying the Shannon entropy measure and computing the mutual information I_1 between A and O yields a counter intuitive result:

$$P1 : I_1(A,O) = -\frac{7}{8}\log\frac{7}{8} - \frac{1}{8}\log\frac{1}{2^{8k}} = k + 0.169,$$
$$P2 : I_1(A,O) = -2^{k+1}\cdot\frac{2^{7k-1}}{2^{8k}}\log\frac{2^{7k-1}}{2^{8k}} = k + 1,$$

Indeed, from a security standpoint, PROG P1 leaves A highly vulnerable to being guessed (e.g., when it is a multiple of 8), while PROG P2 does not (at least for large k).

Smith et. al. [5] and Hamadou et. al. [2] proposed a vulnerability measure (a min-entropy measure instead of classical Shannon entropy measure) in an attempt to resolve this problem. However, we point out that the vulnerability measure may lead to counter-intuitive results on several other programs while, the results based on the classical Shannon entropy measure matches our intuition. Consider programs P3 and P4 below.

PROG P3 Password Checker
if $A = L$ **then**
$O = 1$
else
$O = 0$
end if

and

PROG P4 Binary Search
if $A \geq L$ **then**
$O = 1$
else
$O = 0$

end if

Consider $L = |\mathcal{A}|/2$ is a publicly known program parameter. The intuition is that PROG P4 leaks much more information than PROG P3, because when k is large, the probability of $A = L$ becomes so low that PROG P3 leaks almost no information. But PROG P4 always leaks 1 bit of information, irrespective of $|\mathcal{A}|$. Now, consider the mutual information based on Shannon entropy (I_1) and the vulnerability metric (I_v):

$$\begin{cases} I_v(P3, 2^k) = 1, \ I_v(P4, 2^k) = 1 \\ I_1(P3, 2^k) \approx 0, \ I_1(P4, 2^k) = 1 \end{cases}$$

Hence, the fundamentally challenge is to device an information leakage metric that is intuitive and conflict-free. Towards this goal, we investigate a class of entropy measures called Renyi-entropy [4] – a well known generalization of various entropy measures including, the classical Shannon entropy, min-entropy and guessing entropy, one-guess vulnerability measure, etc. More precisely, Renyi-entropy defines a family of entropy measures based on a parameter $\alpha \in (0, \infty)$ such that $\alpha = 1$ corresponds to the classical Shannon entropy, $\alpha = \infty$ corresponds to the min-entropy and $\alpha = 0$ corresponds to the vulnerability one-guess entropy.

A. Main Results

We informally state our main results below. For detailed claims and proofs please refer our tech-report [6].

Conflicts in Metrics: We show that information flow metrics proposed by past work are inherently prone to *conflicts*, that is, there exists programs P_1 and P_2 and Renyi-entropy parameters α, β ($\alpha \neq \beta$) such that $I_\alpha(P_1) > I_\alpha(P_2)$ and $I_\beta(P_1) < I_\beta(P_2)$. In addition, we show how to construct infinitely many programs wherein different Renyi-entropy based information leakage measures are in conflict.

A conflict-free Metric: For any program P, the asymptote $\theta^{[1]}(I_\infty(P))$ is a conflict-free.

Examples: It is easy to see using the proposed metric $\theta(I_\infty(P))$ one can resolve the conflicts in information leakage metrics for programs P1-P4. In this example, we show the application of our metric to PROG P5.

PROG P5 Modulo

$$O \equiv A \mod L$$

In PROG P5, one can show that the optimal choice of the low input L_i (for the i^{th} program run) is given by:

$$\{L_1^*, \cdots, L_i^*\} = \arg \max_{L_1, \cdots, L_i \in \mathcal{L}} lcm(L_1, \cdots, L_i)$$

where $lcm(L_1, \cdots, L_i)$ refers to the least common multiple [1] of $L_1, \cdots, L_i$. Using our metric one can show that the information leakage in P5 for n runs is given by:

$$I_\alpha(P5, |\mathcal{A}|, n) \quad = \quad \Sigma_{i=1}^{n} \log(L_i^*), \forall \alpha \in (0, \infty)$$

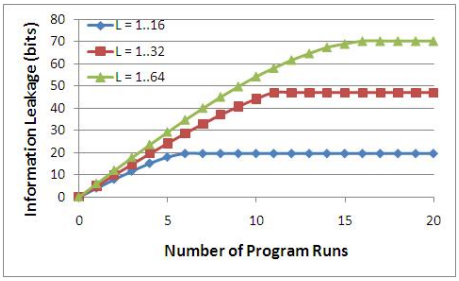

Fig. 1. Quantifying Information Leakage Across Multiple Program Runs

Hence, the information leakage metric for PROG P5 grows linearly with n, the number of program runs as long as $\mathcal{L}$ is sufficiently large. However, for finite $|\mathcal{L}|$ information leakage drops to zero after roughly $\frac{|\mathcal{L}|}{\log_e(|\mathcal{L}|)}$ program runs, where $\log_e$ denotes the natural logarithm. For example, when $\mathcal{L} = \{1, \cdots, 16\}$ then the optimal choice of L_i's is given by $\{16, 15, 13, 11, 7, 3\}$; further choices of L and subsequent program runs do not offer more information about the high input to the adversary. Figure II-A shows the rate of information leakage with the number of program runs for $\mathcal{L} = \{1, \cdots, 16\}$, $\{1, \cdots, 32\}$ and $\{1, \cdots, 64\}$.

III. Limitation

The results presented in this paper applies only when the number of output symbols and the number of program runs is finite and independent of $|\mathcal{A}|$.

Acknowledgements

Research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the author(s) and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorised to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

References

[1] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press, 3 edition, 2009.

[2] Sardaouna Hamadou, Vladimiro Sassone, and Catuscia Palamidessi. Reconciling Belief and Vulnerability in Information Flow. In *IEEE Symposium on Security and Privacy*, pages 79 –92, 2010.

[3] James L. Massey. Guessing and entropy. In *IEEE International Symposium on Information Theory*, page 204, 1994.

[4] A. Renyi. On measures of entropy and information. *Fourth Berkeley Symposium on Mathematical Statistics and Probability*, pages 547–561, 1961.

[5] G. Smith. On the foundations of quantitative information flow. *Foundations of Software Science and Computational Structures*, pages 288–302, 2009.

[6] J. Zhu and M. Srivatsa. Quantifying information leakage in finite order deterministic programs. *ArXiv e-prints*, 2010.