Relating Maxwell's demon and quantitative analysis of information leakage for practical imperative programs

Kushal Anjaria† and Arun Mishra

Department of Computer Science & Engineering, DIAT, Pune 411025, *India*

(Received 28 April 2017; revised manuscript received 20 July 2017; published online 10 September 2017)

Shannon observed the relation between information entropy and Maxwell demon experiment to come up with information entropy formula. After that, Shannon's entropy formula is widely used to measure information leakage in imperative programs. But in the present work, our aim is to go in a reverse direction and try to find possible Maxwell's demon experimental setup for contemporary practical imperative programs in which variations of Shannon's entropy formula has been applied to measure the information leakage. To establish the relation between the second principle of thermodynamics and quantitative analysis of information leakage, present work models contemporary variations of imperative programs in terms of Maxwell's demon experimental setup. In the present work five contemporary variations of imperative program related to information quantification are identified. They are: (i) information leakage in imperative program, (ii) imperative multi-threaded program, (iii) point to point leakage in the imperative program, (iv) imperative program with infinite observation, and (v) imperative program in the SOA-based environment. For these variations, minimal work required by an attacker to gain the secret is also calculated using historical Maxwell's demon experiment. To model the experimental setup of Maxwell's demon, non-interference security policy is used. In the present work, imperative programs with one-bit secret information have been considered to avoid the complexity. The findings of the present work from the history of physics can be utilized in many areas related to information flow of physical computing, nano-computing, quantum computing, biological computing, energy dissipation in computing, and computing power analysis.

Keywords: information, Maxwell's demon, second principle of thermodynamics, information security, reversible system

PACS: 05.70.−a, 05.70.Ce, 07.05.Bx **DOI:** 10.1088/1674-1056/26/10/100507

1. Introduction

From the early days of computers and computing processes, physics and computer fields have a long traditional straddling. The physics field considers computers as an engine. The engine which uses its energy to generate mathematical work and while doing this work it generates heat. This phenomenon can be shown in the equation below:

$$\text{Computational engine energy} = \text{mathematical work} + \text{waste heat} \cdots . \quad (1)$$

In the present paper, the aim is to link the conventional and contemporary methods used for quantitative analysis of information leakage in imperative programs with the physics. For this purpose, the second principle of thermodynamics and Maxwell's demon[1] has been used. Our aim is to show that even if system leaks information, the attacker has to do a certain amount of physical work to obtain leaked information. For this purpose, various experimental setups for Maxwell's demon have been considered.

In physics, simple meaning of information has been considered, i.e., "information is any entity which you do not already know".[2] To measure the information, Claude Shannon came up with the information entropy and information theory in 1948. He suggested that larger the information content, larger the entropy. To find the entropy of information, Shannon considered thermodynamic microstate of a computer system with a certain degree of freedom and came up with information entropy formula.

In computer science, to quantify information leakage in secure imperative programs, the non-interference property is also used along with the information theory. The non-interference policy is confidentiality policy which secure programs satisfy to protect the information. To satisfy non-interference property,[3] the computer system is divided into low-security sensitive input/output entities and high-security sensitive input/output entities. A system satisfies non-interference policy if any sequence of low input will produce the same low output regardless of the high input. Denning[4] first introduced quantitative information flow. Her work revealed that to quantify information leakage, change in uncertainty associated with information at different states needs to be considered. After that, a lot of work have been done to quantify information leakage in imperative programs using Shannon's entropy and non-interference. In Refs. [5] and [6], quantitative analyses of information leakage in abstract computational systems have been studied using Denning's principle. After that Clark, Hunt, and Malacaria Refs. [7] and

†Corresponding author. E-mail: kushal.anjaria@gmail.com

© 2017 Chinese Physical Society and IOP Publishing Ltd

[8] started quantifying information leakage in imperative programs with loops conditions and assignment statements. All the work related to quantification of information leakage in the imperative program was based on Shannon's entropy formula. In some other cases like quantification of leakage in a multithreaded program, non-terminating program, mathematically derived variations of Shannon's entropy formula was used. As a reason, the aim of the proposed work is to go back to the history and try to find the possible setup of Maxwell's demon experiment for the mathematically derived variations of Shannon entropy formulas.

The benefit to revisit Maxwell's experiment for various variation of Shannon entropy formula is it will relate physics and computer again. Solutions of many complex problems related to quantification of information flow can be thought of by physicist as well as computer scientists. Information flow security in quantum computing and nano-computing is also based on the Maxwell demon experiment. The proposed work may open the doors in that direction also.

In the present paper, five types of imperative programs have been derived. For these programs, mathematically and information theoretically derived variations of Shannon entropy formula have been used in literature. For these five types of contemporary imperative programs, the relation between quantitative analysis of information leakage and Maxwell's demon principle will be derived. Not only possible Maxwell's demon experimental setup but also the amount of minimal work required to keep secure content protected from the attacker will also be calculated. In other words, work required by the attacker to obtain the secret content from the imperative programs has been calculated. The five types of imperative programs in which variations of Shannon's entropy formula have bees been used are as follow:

Type 1 Simple imperative program

The quantification of information leakage in the simple imperative program has been described by Clark, Hunt, and Malacaria in Refs. [7] and [8].

Type 2 Imperative multithreaded program

To quantify information leakage in the imperative multithreaded program, scheduler effect also needs to be considered. In Refs. [9], [10], and [11] information leakage has been quantified in the imperative multithreaded program.

Type 3 Point to point information leakage in the imperative program

This method was first proposed by Tom Chothiya *et al.* in Ref. [12]. In this paper, a method has been proposed to find leakage between any two arbitrary points in the imperative probabilistic program. This method makes convenience to measure specific leaks in the program and it can be used to investigate information security of specific portion of the program.

Type 4 Information leakage of non-terminating processes

In some programs or systems, secret contents are continuously generated as they execute repeatedly. In Ref. [13], Biondi *et al.* have quantified information leakage in such programs where secret content is infinite.

Type 5 SOA-based web services

In this method, loosely coupled components provide services to other component and in this way information flow is generated. Thus, in SOA-based web services, interference is not only between high-security sensitive content and low-security sensitive content but also it is among secure content and low-security sensitive content and other architectural parameters.[14]

The motivation for this work has been explained in the subsection below.

1.1. Motivation and aim

Our motivation for the present work is in two-dimensional, one dimension is about computational physics and the second dimension is related to non-interference policy. In computational physics, two types of devices are described, i.e., logically reversible devices and logically irreversible devices. In the logically reversible devices, the inputs of the devices can be uniquely identified by observing their output, while in logically irreversible devices; inputs cannot be uniquely determined by observing their output. Some of the examples of logically irreversible devices are Boolean operations like AND, OR, and XOR.[15]

The physical devices relate observable outputs and inputs; similarly, non-interference policy relates observable (low-security sensitive contents) and secure content. In simple words, the attacker should not be able to know the secure content by observing output or observables of the system. Thus if we relate physical computation and non-interference, we can derive that the aim of the non-interference property is to convert the system or program into the irreversible system. But as described in Ref. [16], it is not possible to satisfy non-interference property completely in any system. At some point, low-security sensitive content and high-security sensitive content will always interfere. That means in computational physics terminology, at some point system will become logically reversible device and it can violate the second principle of thermodynamics as suggested by Rolf Landauer in 1961 in Ref. [17]. Landauer suggested that these "reversible" measurements could be used to sort the molecules and violating the Second Law. Our aim in this paper is to observe this logically reversible device when Maxwell demon experiment and Landauer's principle is applied to them. And through this observation, study the non-interference property from a physical

aspect. Thus, in short, the non-interference policy and Shannon's entropy both are related to physics in one or the other way, that is, the hint which encourages us to study quantitative analysis of information leakage of various systems from thermodynamics and physics aspects. In the present work, the phenomenon of the reversible and irreversible systems has been in relaxed comparison to quantum computing. In quantum computing, the system is reversible only if all the prior states are physically achievable. In the present work even if the attacker is logically able to get the initial variable value or state after the execution of program then the program is considered as reversible in the present work.

1.2. Organization of paper

Section 2 of the present paper incorporates preliminary concepts required to understand the paper. Section 3 contains information about Maxwell's demon and the relation between Shannon's entropy formula and the second principle of thermodynamics. In Section 4, Maxwell demon experimental setup for five types of imperative programs is discussed. 1) The simple imperative program, 2) imperative multithreaded program, 3) imperative program with a point to point information leakage, 4) imperative program which generates infinite leakage or observable, and 5) SOA implementation of the imperative program. In Section 4 physical work required to obtain secret from attacker's perspective is also calculated. Section 5 provides a review of literature related to physics, information theory, and computation. In Section 6 conclusion and possible future work in the direction of physics and quantitative analysis of information leakage has been discussed.

2. Preliminaries

In this section, some of the concept relating physics and information are included. The physics and information theory is associated with each other because of the second principle of thermodynamics. The second principle of thermodynamics associates state function known as entropy S and states when the system undergoes transformation.[18] In simple words, the second principle of thermodynamics states that any system will absorb heat at temperature say T, and then if the system is irreversible then it will dissipate a certain amount of energy which can be used to do work.

Thus as per the second law of thermodynamics,

$$\Delta S \geq \frac{\delta Q}{T}, \tag{2}$$

where ΔS is changed in entropy of the system; T is temperature and δQ is heat. Shannon considered Eq. (2) with microstates and with degrees of freedom and got entropy formula of information as shown in Eq. (3) below:

$$S = -\sum P \log P, \tag{3}$$

where P is uncertainty associated with the state of the system and S is the entropy of information. From Shannon's formula shown in Eq. (3), physical work in terms of computing can be described[18] as:

$$W = S(h|P). \tag{4}$$

In Eq. (4), S is Shannon's entropy, h is secure information, and P is observables which are probabilistically related to inputs. Equation (4) also hints that physical work and Shannon's entropy can be related.

From Eq. (2), Maxwell bought very interesting concept which is known as Maxwell's demon which can relate quantification of information and energy dissipation. This concept has been explained below.

3. Maxwell's demon and information quantification

In a thought experiment, Maxwell considered a system with two neighboring chambers at the same temperature initially. The demon is an intelligent agent who can observe the molecules of gas in both the chambers. In both the chambers, some of the molecules will be moving faster than the average while some will be slower. The demon will be able to open and close the molecule size trapdoor available in the partitioning wall of two chambers and collect faster (hot) molecules in one chamber and slow (cold) particles in the other chamber. Maxwell suggested that this temperature difference can be used to create useful work. Based on Maxwell's demon, simplified one molecule model was introduced in Refs. [19] and [20]. From a computational perspective, Bennet's model of Maxwell's demon has been described in Fig. 1.

In Fig. 1 as shown in level-1, the demon is completely ignorant about the position of a gas molecule in the closed system. The system is connected to a heat reservoir at temperature T_0. From information security perspectives, the demon can be considered as an attacker and system can be considered as a computer program. The secret information in the system is a place of the particle in the system. In level-2, the demon inserts partition without any cost in the closed system which divides the system into two chambers. Now demons know that the particle is either in the left chamber or in the right chamber. From computer science perspective it can be said that attacker is ready to execute the program now. In level-3, the demon is ready to insert the piston into the system without any cost. It is assumed that piston is inserted in the system till partition and still whereabouts of the particle is unknown. This is similar to the situation in the computer where the attacker knows that computer program is divided into low and high-security sensitive content but does not know anything about

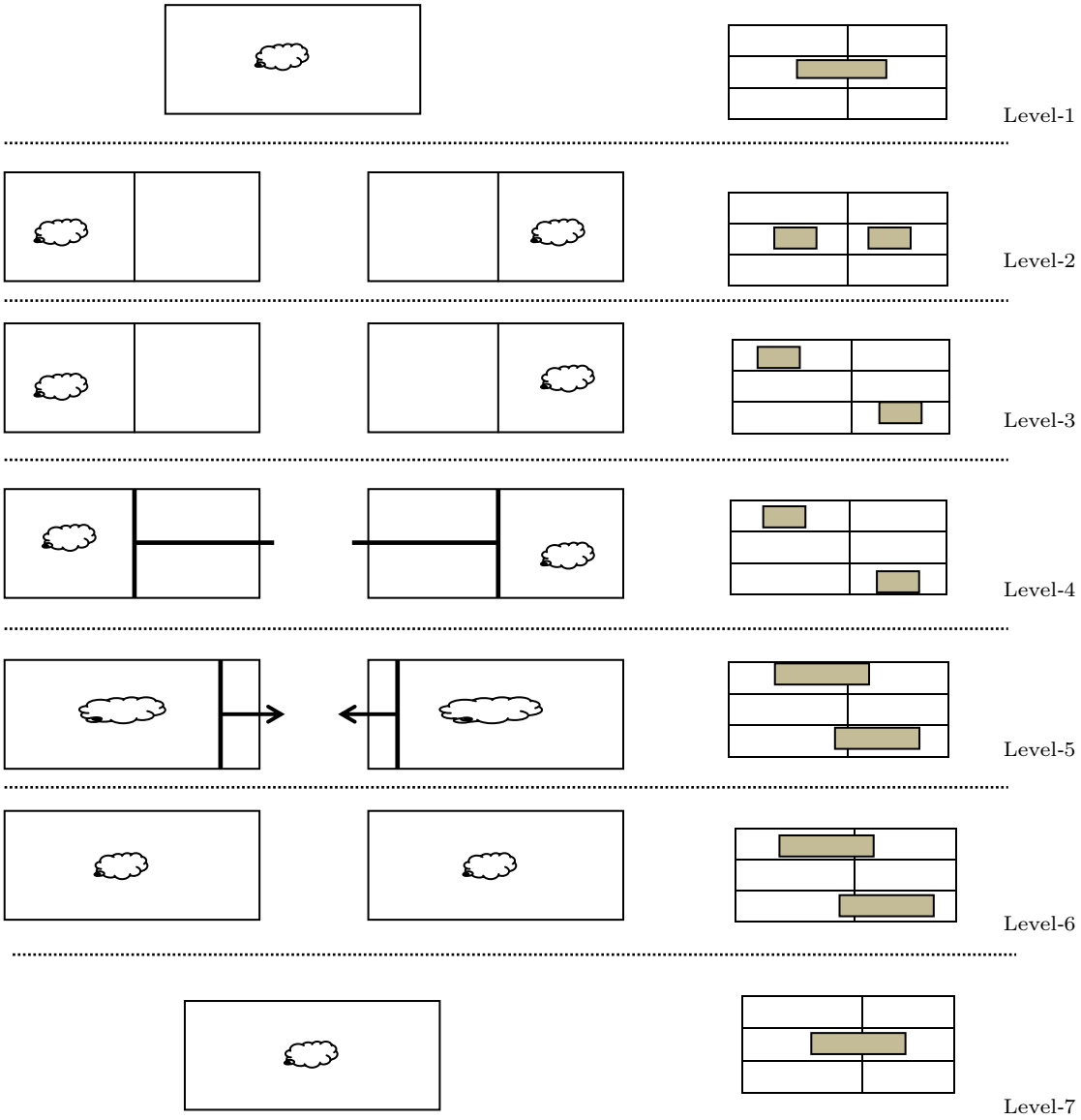


Fig. 1. Relation between information entropy and physical entropy.

the high-security content before execution. The information about the whereabouts of the particle in the system has been compared with the high-security sensitive content. In level-4, demon pushes the piston isothermally till the end of the chamber say towards the left side. To push the chamber, demon requires physical work $k_{\mathrm{B}}T\ln 2$, where k_{B} is Boltzmann constant and T is temperature. In level-5, the demon will remove the partition in the chamber without any cost. The situation shown in level-4 and level-5 can be compared with the situation in computer science when attacker executes the program and waiting for the outcome.

Due to the expansion, in level-6, the system will come to its original state. If the particle was on the left side then, the gas molecule will do the work and heat will flow into the reservoir. If the particle is on the right of the piston, then heat will flow from reservoir.[21] By observing heat of the reservoir, the attacker will be able to know that whether the particle was in the right chamber or the left chamber. In computer science terminology, it is similar to the situation where the attacker knows about the secret content by observing the outcome of the program after program execution due to a violation of non-interference. Because of this interference, previous state of high-security sensitive content will be known to the attacker by observing the output. That means, in the physical term, the system will be reversible.

Thus, Maxwell's demon shows that how one-bit secret information can be leaked. Landauer suggested from Maxwell's demon that to reset the information gained by an attacker, which means to make situation level-1 = level-7, work $K_{\mathrm{B}}T\ln 2$ is required. Thus, Landauer principle calculates work required to keep information secret in the computer system. Thus to gain the secret attacker needs to do more than $K_{\mathrm{B}}T\ln 2$ work.

In Ref. [22], a similar experiment was done by including weight in the system to measure the work done by the system. Figure 2 explains the experiment with weight. The figure

below explains that how an attacker can use the information about the position of the particle for his/her benefit. The attacker can, for example, use the system as an engine. He can add a partition to the left of the particle and can perform useful work for his/her benefit.

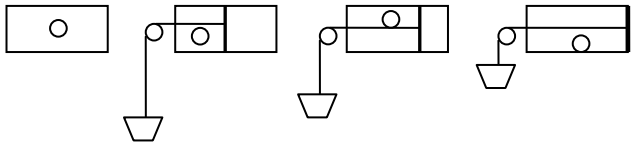


Fig. 2. One molecule engine proposed by Szilard.

In Ref. [23], Bennett represented this concept using position versus potential graph. The graph is shown in Fig. 3 below. It describes that to know the position of the particle, a certain amount of potential barrier needs to be overcome.

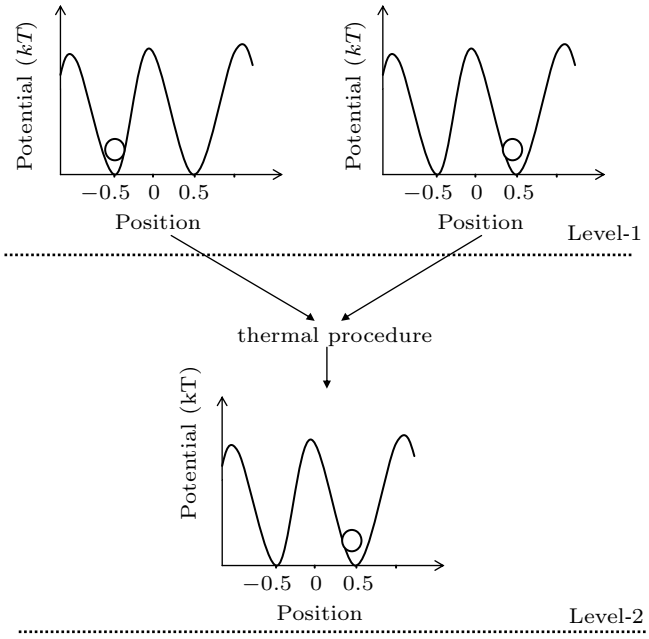


Fig. 3. Position versus potential graph to understand Maxwell's demon.

As shown in level-1 of Fig. 3, the attacker does not know the position of the particle. So particle can be either left at position -0.5 or at right with position $+0.5$. Between both the position potential barrier of say $k_{\mathrm{B}} \ln T$ is available. So initial entropy, S_{i} is $k_{\mathrm{B}} \ln T$. After that to know the position attacker does the work equal to $k_{\mathrm{B}} \ln T$. Due to this, as shown in level-2, the attacker knows the position of the particle. In the case shown in level-2, the position of the particle is to the right of the partition. For the situation shown in level-2, final entropy S_{f} is 0. Thus from levels 1 and 2, entropy difference can be calculated as shown below:

$$\Delta S = S_{\mathrm{f}} - S_{\mathrm{i}} = 0 - k_{\mathrm{B}} \ln T = -k_{\mathrm{B}} \ln T. \tag{5}$$

From Eq. (5), the secret behind the negative sign in Shannon's entropy formula can be understood by the computer scientists.

In the present work, we will demonstrate possible settings of Maxwell's demon experiment for a practical imperative program like the simple imperative program, imperative multithreaded program, a probabilistic program with a point to point information leakage program, a program which generates infinite secret, and the program executes in SOA based architecture.

4. Maxwell's demon experimental setup for practical imperative programs

The quantitative analysis, based on Shannon's entropy formula has been used widely these days to decide the threshold of the information leakage in the programs. The Shannon's formula is derived from the second principle of thermodynamics and Maxwell's demon experiment. So our aim in this paper is to go in the reverse direction and try to find how the experimental setup of Maxwell's demon can be, for practical programs which have the secret of one bit. In this section, Maxwell demon's experimental setup will be investigated for five types of program:

1) Simple imperative program;

2) Imperative multithreaded programs;

3) Probabilistic programs with point to point information leakage;

4) Imperative program which generates infinite secret;

5) The program executes in SOA based architecture.

4.1. Simple imperative program

In the present work, the simple imperative program has been considered as a sequential program without loops. An example of simple imperative program snippet is as below:

secret variable $= h$

$o := h$.

As shown in the snippet, h is high-security sensitive variable and o is observable. Assume that high-security sensitive variable h and observable o are one bit variable. In the second statement, the non-interference property is violated as high-security sensitive variable and observable interfere with each other. Thus, due to the second statement of the snippet, the program becomes reversible and violates the second law of thermodynamics as suggested in Maxwell's demon experiment.

For these types of simple imperative programs, the setup of Maxwell's demon experiment will be the same as shown in Fig. 1. The reason is, in the program, the one-bit secret is there which can be compared with the secret position of a gas particle in Maxwell demon experiment. The imperative program can be compared with the closed system as described in Maxwell's experiment. Work required to protect the secret content will be $k_{\mathrm{B}} \ln T$. In other words, the attacker needs to do minimum work $k_{\mathrm{B}} \ln T$ to gain the secret.

4.2. Imperative multithreaded program

In simple terminology, multithreading is the ability of processor with a single core or multiple cores to execute more than one thread using the scheduler in such a way that it appears to the user that threads are being executed concurrently. In the present paper, it has been assumed that multithreaded programs are executed on a single core or a single processor with the help of scheduler. In Refs. [9], [10], and [11] methods to quantify information leakage are provided. The scheduler in multithreaded program assigns the probability to the thread for execution and based on probability it picks up a thread for execution. Information leakage in the multithreaded program depends on the sequence in which threads are picked up by the scheduler for the execution. An example of the imperative multithreaded program is as follows:

$$
\begin{aligned}
&h = h \mod 2;\\
&Th_0 :\\
&\qquad l = h;\\
&Th_1 :\\
&\qquad Th_{10} :\\
&\qquad\qquad l = k;\\
&\qquad Th_{11} :\\
&\qquad\qquad l = m
\end{aligned}
$$

As shown in the example, h is high-security sensitive variable while l, k, and m are low-security sensitive variable. It has been assumed here that all variable, i.e., h, k, l, and m are one-bit variables. For this imperative multithreaded program, the scheduler will assign the probabilities to threads Th_0 and Th_1. The threads Th_{10} and Th_{11} are child threads. When $l = h$ is executed at that time the non-interference property will be violated.

From physics perspectives, if we want to design Maxwell's demon experiment then effects of scheduler need to be considered. To consider scheduler effect, probability needs to be included in Maxwell demon experiment. The probability was included in this experiment by Malacaria and Smeraldi in Ref. [18]. In the experiment Malacaria and Smeraldi assumed that there is more than one system in the experiment and particles are found in two chambers of the systems with probability μ_1 and μ_2. It has been assumed that $\mu_1 > \mu_2$ and $\mu_1 + \mu_2 = 1$. The boxes or systems considered in Fig. 4 have unit volume. In physics terminology, it was assumed that all the systems are identical and contain an ideal gas molecule. It has been assumed that piston is on the center position of the systems and shafts are connected at the end. Since molecules are in both the chambers of the systems, it will hit the piston from both the sides. The concept is shown in Fig. 4 below. In the figure, two systems are shown with the probability that particles found in the left chambers (marked with label 'a' in Fig. 4) of both the system with probability μ_1 and on the right chambers (marked with label 'b' in Fig. 4) of both the system with probability μ_2. So piston will press the gas particles in the right chamber or in the left chamber. This concept is similar to the thread scheduling done by the scheduler.

As shown in Fig. 4, pistons will move towards right till the pressure on both the sides is equal. This can be used to extract work from the system. In the experiment, $P_i = \mu_i k_B T$. Here P_1 is the pressure in left chamber and P_2 is the pressure in the right chamber. The volume of chambers are unit volume so $V_1 + V_2 = 2$. The expansion takes place till pressure in both sides are equal so, $\mu 1/V1 = \mu 2/V2$. That means at the end of expansion we have $V_1 = 2\mu_1$.

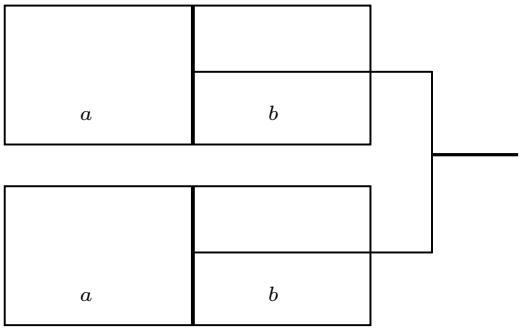


Fig. 4. Maxwell's demon experiment with two systems and probabilities.

Thus, from thermodynamics perspectives, work done by this system can be calculated as

$$
\begin{aligned}
W_{\rm exp}/k_B T &= \frac{1}{k_B T}\int_1^{2\mu_1} (P1 - P2)\,{\rm d}V\\
&= \int_1^{2\mu_1} \frac{\mu_1}{V}\,{\rm d}V - \int_1^{2\mu_1} \frac{\mu_2}{2-V}\,{\rm d}V\\
&= \mu_1 \ln 2 + \mu_1 \ln \mu_1 + \mu_2 \ln 2 + \mu_2 \ln \mu_2\\
&= \ln 2 - H(\mu_1, \mu_2) \ln 2. \qquad (6)
\end{aligned}
$$

Thus, the attacker needs this much work to know about the secret in the multithreaded program which executes under the effect of scheduler. Similarly, for n systems, the above case is generalized and calculated work done is provided below. Detailed calculation of work done by n-state system is provided in Ref. [18]. It is

$$M_n \ln n/M_n - M_{n-1} \ln n - 1/M_{n-1} + \mu_n \ln \mu_n. \qquad (7)$$

4.3. Imperative program with point to point information leakage

In this method of quantification, the simple imperative program has been considered but leakage has been calculated between any two arbitrary points in the imperative program. In Ref. [12], this type of information leakage model has been discussed. In the paper, leakage in practical JAVA programs has been quantified. One similar example of an imperative program is given below:

```
Program-1
    Statement-1;
    Statement-2;
    Statement-3;
    Statement-4;
    Statement-5;
End
```

As shown in the program there are five statements. Aim here is to find leakage between any two statements. The interference between high-security sensitive variable and low-security sensitive variable may be or may not be between selected arbitrary points of the program. For example, in the above code snippet, interference is in statement-3. If statement-1 and statement-2 are selected to quantify leakage then it is possible that quantified leakage is zero. In the above program, it is assumed that high-security sensitive variable is of one bit.

The setup of Maxwell's demon experiment for this program will be hypothetical in nature. The setup is shown in Fig. 5 below. In program two arbitrary points can be selected. Similarly, as shown in figure system walls XY and PQ are movable and allow attacker or user to select two points in the system. A particle of an ideal gas is in the left chamber as shown in the figure. The wall XY is a movable wall. If it is moved towards the right and crosses the particle then it is assumed that particle will be outside of the system through the wall XY. This is the hypothesis considered in this setup.

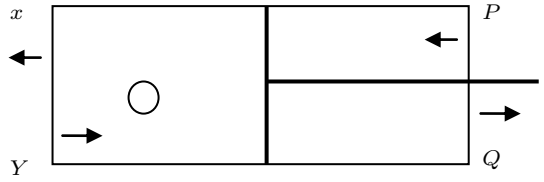


Fig. 5. Experimental setup of Maxwell's demon experiment to quantify point to point information leakage in imperative program.

As shown in the figure, volumes of chambers are V_1 and V_2. In the previous case, unit volumes were considered for both the chambers, but here this will not be the case. So in this case, $V_1+V_2 \neq 2$ but let us assume that $V_1+V_2 = V_{\text{total}} = x$. It has been assumed that the piston will cover volume x–y during expansion. The probability that the particle is within the system boundary decided by the user is μ. Let us assume that the particle is within the system then work done by this system can be calculated as

$$\begin{aligned} W_{\text{exp}} &= \int_x^{x-y} P\,\mathrm{d}v \\ &= \int_x^{x-y} \frac{\mu}{V}\,\mathrm{d}v \\ &= -\mu(\ln y - \ln(x-y))k_{\text{B}}T. \end{aligned} \tag{8}$$

Thus, the attacker needs to do minimum work $-\mu(\ln y - \ln(x-y))k_{\text{B}}T$ as shown in Eq. (8).

4.4. Imperative program which generates infinite secrets

The quantitative analysis successfully measures the protection of secure content in the system but till 2014, it was assumed that secret and observations produced by the system are finite. But in 2014, Biondi *et al.* suggested in Ref. [13] that there can be programs or systems which can generate infinite secret and observables. In Ref. [13] method to quantify information leakage in such program was also provided. For example, consider the code snippet from Ref. [13] is provided below.

```
secret int1 h;
  if (h == 0) then
        observable int1 o := 0;
      while (0 == 0) do
        random o := randombit
            (0.5);
        od
      else
        observable int1 o := 0;
      while (0 == 0) do
        random o := randombit
            (0.75);
        od
  fi
  return;
```

In the code snippet, variable h is high-security sensitive variable. If the value of variable h is zero then program will produce a string of infinite zeros and ones with same probability, i.e., 0.5. If the value of h is 1 then zero will be produced with probability 0.75 and 1 will be produced with probability 0.25. Thus as shown in the code snippet, it generates infinite observable. The attacker will observe the behavior of some observable and based on that he will try to guess the value of high security sensitive variable h. It is clear for the above code snippet that after some iteration, leakage will be entire variable h, i.e., 1 bit.

The setup of Maxwell's demon experiment for programs which generate infinite observable is shown in Fig. 6 below. In the figure, the probabilistic observable generation is omitted for simplicity. The probabilistic system will be similar to the one shown in Subsection 4.2. The figure shows the experimental setup to deal with infinite observables.

As shown in the figure, hypothetical experimental setup of Maxwell demon experiment needs to be considered in this case. Both the chambers shown in Fig. 6 are infinite in length.

The piston is in the center of the system and in this experiment, expansion of piston will take place till significant observable work is not achieved by the system. Assume that attacker needs k observables to find where the particle is.

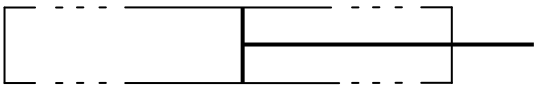


Fig. 6. Experimental setup of Maxwell's demon experiment to quantify information leakage in imperative program which generates infinite observables.

Here volume of both the chamber is infinite. Thus $V_1 + V_2 = V_{\text{total}} = \infty$. It is assumed that piston moves till attacker gets significant knowledge. So the assumption is expansion takes place till volume m. Work done in the system can be calculated as

$$\begin{aligned} W_{\text{exp}} &= \int_{\infty}^{m} P\,\mathrm{d}v \\ &= \int_{\infty}^{m} \frac{1}{V}\,\mathrm{d}v \\ &= \ln(\infty) - \ln(m) k_{\text{B}} T \\ &= -(\ln(m) - \lim_{k\to\infty}(k)) k_{\text{B}} T. \end{aligned} \tag{9}$$

At volume m, an attacker may know the secret. In Ref. [13] equation $H(h) - \lim_{k\to\infty} H(h|_{o_1,o_2,\ldots,o_k})$ is used to quantify leakage. This is similar to Eq. (9).

4.5. Program executes in Service Oriented Architecture (SOA)

SOA is an architectural solution which provides loose coupling of services through the orchestration of service components. In the case of an SOA-based program, information leakage will not be just because of interference among the low-security sensitive and high-security sensitive variables but other architectural parameters will also play a crucial role in the quantification of information leakage. The detail about the SOA and attacks related to SOA are discussed in detail in Refs. [24] and Refs. [14] and [25] respectively. In SOA, the parameters like network link will be outside of the closed system. Thus in the SOA-based web services and programs, non-interference property needs to be considered in a global way. In this SOA-based structure, interference is not only between the high variable and low variables but all other architectural factors. In the SOA-based systems, the architectural parameters like the effect of link failure also needs to be considered which interfere with secure content.

The setup of Maxwell's demon experiment will be very similar to the one shown in Figs. 1 and 2. But as other factors (like network link) also get involved in the SOA-based program, in Maxwell's demon experiment also other factors need to be considered. So Maxwell's demon experimental setup, it is considered that the system is not in thermal isolation. The temperature outside the system can affect the work done by the system. The environmental elements affect the system; this phenomenon is known as decoherence. In SOA system, the attacker should know about the frequency of the link, similarly, in Maxwell's demon experiment, the demon should know about the temperature outside the system. This situation rarely takes place in both the cases. But avoidance of decoherence situation is big research challenge in conventional computing as well as quantum computing. In this situation, calculation of work done is very complex to calculate. Recently, Anjaria and Mishra, in Ref. [26] have successfully demonstrated the quantification of information leakage in SOA-based system.

From analysis of various experimental setups it can be concluded that at present, information security policies based on non-interference policy of various types of programming scenario were just an extension of Maxwell demon experimental setup. But for cloud computing and SOA-based system, this conventional variations of Maxwell demon experimental setup will not work due to decoherence. To eliminate decoherence, computer scientists and physicists have to do combined efforts. In Ref. [26], Anjaria and Mishra have considered non-interference policy globally and quantified the leakage. This can be considered as a way to mitigate decoherence.

5. Related work

From the second law of thermodynamics, in 1879, Rudolph Clausius[27] suggested that spontaneous processes have preferred direction and this direction is from hotter body to cooler body. In deriving information theory and entropy, Shannon took help of this fact represented by Rudolph. Shannon first believed that information and the second law of thermodynamics can be related but then he focused on very important questions: Just like mechanical systems, is it possible to get knowledge about the work done by the system by observing states of the system only? In the question, work done can be considered as minimal work required protecting the secret content. The answer to the above question was found from the concept of Maxwell's demon. This famous concept of Maxwell's demon was represented by Maxwell. The history of the experiment is available in Ref. [28].

Lio Szilard converted Maxwell's demon experiment into single molecule based Maxwell's demon experiment in Ref. [19]. This same single molecule-based experiment has been used in the paper to explain the relation between quantitative information and thermodynamics. Like Maxwell, Szilard also represented thought experiment only. In 1961, Landauer

suggested in Ref. [17] that erasure of information in single molecule system or single bit information system is a dissipative process. In Ref. [29], Berut *et al.* represented actual experimental set up to verify Szilard and Landauer's principle. In Ref. [29], Berut *et al.* used a system of a single colloidal particle, trapped in a modulated double-well potential to verify thought experiment represented by Szilard and Landauer.

The logical explanation for the presence of demon and why demon will not be able to break the second law of thermodynamics was presented by Bennett and Oliver Penrose in Refs. [23] and [30] respectively. Bennett and Oliver Penrose argued that presence of demon and resetting of demon's memory to initial place will dissipate heat. Landauer suggested in Refs. [17] and [31] that resetting of memory always dissipate heat. From Landauer's argument, Bennett and Oliver calculated and explained that due to this heat dissipation, the demon will not be able to violate the second principle of thermodynamics. Thorough and state of the art survey of thermodynamics and information is discussed in Ref. [2] by E Lutz and Sergio C State of the art survey of the relation between thermodynamics and information in provided in Ref. [32] by Juan *et al.*

The Maxwell's demon, Landauer and Bennett's principles have been widely used in mechanics, computer science, and statistics. In Ref. [33], John T and Craig L provided quantum dot cellular automata model of Maxwell's demon. This automata model is very useful to study computation and energy dissipation in computing. In Ref. [34], Mandal and Jarzynski proposed information processing solvable model of Maxwell's demon which can be used in computation and mechanics widely to model the work and energy dissipation. In Refs. [18] and [35], Malacaria and Smeraldi successfully attempted to relate secure computing, confidentiality, quantitative analysis of information and physical process and systems. Malacaria and Smeraldi argued using thermodynamics that any computing system whose final state is observable must dissipate $Wk_{\mathrm{B}}T\ln 2$ energy. In equation, W is information theoretic notion of quantitative information flow. In Refs. [18] and [35] timing channel and entropy variations were also addressed. The work done by Malacaria and Smeraldi is the main motivation behind the present paper.

6. Conclusion and future scope

Once the fundamental relationship was established between information and thermodynamics, an attempt has been made in the present paper to relate recent trend in confidentiality and information quantification with thermodynamics. In the present work, minimal work required by an attacker to gain confidential information is also calculated for five cases, i.e., information leakage in imperative program, imperative multithreaded program, point to point leakage in imperative program, imperative program with infinite observation, and imperative program in the SOA-based environment. The possible experimental setup of Maxwell's demon was also described for each of the five cases. Although at first possible experimental setups of Maxwell's demon look impractical, in Ref. [29], the simple experimental setup has already been proposed. So in future, setup of other variations of this experiment may also be possible.

At present, we make no claims about practical implications of the proposed method in any specific field but in future, this work will be helpful to find many unanswered questions in the fields of computation energy dissipation, quantitative analysis, and information secrecy. The present work will also help to understand confidentiality in quantum computing. The work can be related in the future with nano-computing and technology, molecular, and biological computing fields. This work bridges the computer science and physics so that physicist and computer scientists can together solve some complex problems in information flow security and quantitative analysis of information leakage.

References

[1] Maruyama K, Nori F and Vedral V 2009 *Rev. Mod. Phys.* **81** 1

[2] Lutz E and Ciliberto S 2015 *Physics Today* **68** (9)

[3] Boudol G and Castellani I 2002 *Theor. Comput. Sci.* **281** 109

[4] Denning D E 1976 *Commun. ACM* **19** 236

[5] Gray III J W 1992 *J. Comput. Secur.* **1** 255

[6] McLean J 1990 *IEEE Proceedings in Computer Society Symposium on Research in Security and Privacy*, pp. 180–187

[7] Clark D, Hunt S and Malacaria P 2007 *J. Comput. Secur.* **15** 321

[8] Clark D, Hunt S and Malacaria P 2002 *Electronic Notes in Theoretical Computer Science* **59** 238

[9] Chen H and Malacaria P 2007 in *ACM Proceedings of the* 2007 *Workshop on Programming Languages and Analysis forSsecurity* pp. 31—40

[10] Ngo T M and Huisman M 2013 *arXiv preprint* arXiv: 1306.2693

[11] Boreale M 2009 *Information and Computation* **207** 699

[12] Chothia T, Kawamoto Y, Novakovic C and Parker D 2013 in *IEEE* 26*th Computer Security Foundations Symposium*, pp. 193–205

[13] Biondi F, Legay A, Nielsen B F, Malacaria P and Wasowski A 2014 in *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science*

[14] Kaur D and Kaur P 2016 *Procedia Computer Science* **78** 298

[15] Bennett C H 1973 in *Maxwell's Demon, Entropy, Information, Computing*, pp. 197–204

[16] McLean J, Millen N J and Gligor V 2001 in*Proceedings of the IEEE Workshop on Computer Security Foundations* pp. 237—238

[17] Landauer R 1961 *IBM J. Res. Develop.* (International Business Machines) **5** 183 doi:10.1147/rd.53.0183

[18] Malacaria P and Smeraldi F 2013 *Information and Computation* **226** 76

[19] Szilard L 1964 *Behavioral Science* **9** 301

[20] Toyabe S, Sagawa T, Ueda M, Muneyuki E and Sano M 2010 *Nat. Phys.* **6** 988

[21] Ladyman J, Presnell S and Short A J 2008 *Studies in History and Philosophy of Science Part B: Studies in History and Philosophy of Modern Physics* **39** 315
[22] Koski J V, Maisi V F, Pekola J P and Averin D V 2014 *Proc. Nat. Acad. Sci.* **111** 13786
[23] Bennett C H 1982 *Int. J. Theor. Phys.* **21** 905
[24] Erl T 2005 *Service-oriented architecture: concepts, technology, and design* Pearson Education India
[25] Yamany H F E, Capretz M A and Allison D S 2009 In *IEEE Congress on Services-I* pp. 653–660
[26] Anjaria K and Mishra A 2017 "Quantitative analysis of information leakage in service oriented architecture based web services", *Kybernetes*, **46** (3)
[27] Clausius R 1879 *The mechanical theory of heat*, Macmillan
[28] Lef H S and Rex A F 2003 *Maxwell's Demon 2: Entropy, Classical and Quantum Information. Computing* (Bristol and Philadelphia: Institute of Physics Publishing)
[29] Bérut A, Arakelyan A, Petrosyan A, Ciliberto S, Dillenschneider R and Lutz E 2012 *Nature* **483** 187
[30] Penrose O 2005 *Foundations of statistical mechanics: a deductive treatment*, Courier Corporation
[31] Landauer R 1961 *IBM J. Res. Develop.* **5** 148
[32] Parrondo J M, Horowitz J M and Sagawa T 2015 *Nat. Phys.* **11** 131
[33] Timler J and Lent C S 2003 *J. Appl. Phys.* **94** 1050
[34] Mandal D and Jarzynski C 2012 *Proc. Nat. Acad. Sci.* **109** 11641
[35] Malacaria P and Smeraldi F 2012 *IEEE 25th Computer Security Foundations Symposium*, pp. 280–290