Information Leakage of Non-Terminating Processes

Fabrizio Biondi, Axel Legay, Bo Friis Nielsen, Pasquale Malacaria, Andrzej Wasowski

Information Leakage of Non-Terminating Processes*

Fabrizio Biondi[1], Axel Legay[1], Bo Friis Nielsen[2], Pasquale Malacaria[3], and Andrzej Wąsowski[4]

1 **INRIA Rennes, France; {fabrizio.biondi,axel.legay}@inria.fr**
2 **Technical University of Denmark, Denmark; bfn@imm.dtu.dk**
3 **Queen Mary University of London; p.malacaria@qmul.ac.uk**
4 **IT University of Copenhagen, Denmark; wasowski@itu.dk**

Abstract

In recent years, quantitative security techniques have been providing effective measures of the security of a system against an attacker. Such techniques usually assume that the system produces a finite amount of observations based on a finite amount of secret bits and terminates, and the attack is based on these observations. By modeling systems with Markov chains, we are able to measure the effectiveness of attacks on non-terminating systems. Such systems do not necessarily produce a finite amount of output and are not necessarily based on a finite amount of secret bits. We provide characterizations and algorithms to define meaningful measures of security for non-terminating systems, and to compute them when possible. We also study the bounded versions of the problems, and show examples of non-terminating programs and how their effectiveness in protecting their secret can be measured.

1998 ACM Subject Classification D.4.6 Security and Protection; G.3 Stochastic Processes; H.1.1 Systems and Information Theory

Keywords and phrases Quantitative information flow, Markov chain, information leakage, infinite execution

Digital Object Identifier 10.4230/LIPIcs.xxx.yyy.p

1 Introduction

Information-theoretical quantitative security techniques evaluate the effectiveness of a system in protecting a secret it depends on. Given a known finite size of a secret in bits, they quantify how many bits of the secret can be inferred by an attacker able to observe the system's output. This value is referred to as *information leakage*, or just leakage. Leakage quantification techniques have been successfully applied to security problems, including proving the effectiveness of bug fixes to the Linux kernel [10], quantifying anonymity [6], and analyzing side channel attacks to the cache of a processor [11].

The theory behind these techniques commonly assumes that the program under analysis terminates at some point, and the computed leakage corresponds to the amount of information that the attacker gains at program termination time. When the secret does not change during computation, a program can be modeled as a *channel matrix*, assigning the conditional probability of each possible output for each possible input. The size of the channel matrix is usually exponential in both secret and output size. We have previously proposed the use of Markovian models instead to overcome this problem [4].

* The research presented in this paper has been partially supported by the MEALS project and by the MSR-INRIA project "Privacy-Friendly Services and Apps".

Markovian models can also be conveniently used to model non-terminating processes, something finite-size channel matrices cannot do. This allows us to study leakage properties of systems like webservices, server modules and operating system daemons.

In this paper we provide techniques and algorithms to quantify the Shannon leakage and leakage rate of non-terminating processes. Shannon leakage has a clear operational significance related to the number of attempts that an attacker has to try to guess a secret [13]. Other measures exist, modeling different security properties (e.g. [16]). Our contributions are:

- We characterize program-attacker scenarios according to the finiteness of the system's secret and the finiteness of the attacker's observation. We show how this characterization influences the finiteness of the information leakage in the scenario.
- We provide a method to compute information leakage of a scenario with either an infinite secret or an infinite observation. Such scenarios cannot even be modeled with a finite size channel matrix. We demonstrate this method with an example.
- We provide a method to compute the rate of information leakage per time unit, when the leakage itself is infinite. This is the case when a scenario has an infinite secret and an infinite observation, as is common e.g. in webservices. We demonstrate this method using a mix node as an example.
- We provide an algorithm to compute how much information is leaked from a given time to another given time.
- We show that determining the exact time in which a given amount of information is leaked is hard, by reduction to the knowingly hard to decide Skolem's problem.

Secret	Observation	Leakage	Leakage rate	Reference
Finite	Finite	Finite	0	Paper [4]
Finite	Infinite	Finite	0	Section 3
Infinite	Finite	Finite	0	Section 3
Infinite	Infinite	Pot. infinite	Finite	Section 4

Figure 1 Leakage of various process topologies

We distinguish four possible scenarios, according to whether the observation by the attacker is finite or infinite and whether the secret itself is finite or infinite. The cases are summarized in Fig. 1. The case with finite observation over a program depending on a finite secret is the terminating case we considered previously [4], while the others will be considered in this paper. When only one of observation or secret is finite the leakage is finite but cannot be computed using the method we introduced previously [4], thus we provide a new technique in Section 3. When both observation and secret are infinite, the leakage is potentially infinite. In this case we compute the rate of leakage, i.e. the amount of information leaked for each time unit. Intuitively, this quantifies the average amount of information the attacker infers for each time unit over an infinite time. This is presented in Section 4. In Section 5 we analyze how much information is leaked in a given time frame and how much time it takes to leak a given amount of information. Section 6 concludes the paper and discusses related work.

2 Background

We refer to literature [8] for the definitions of sample space S, probability of event $P(E)$ and so on. $\mathcal{X}$ is a *discrete stochastic process* if it is an indexed infinite sequence of discrete random variables $(X_0, X_1, X_2, ...)$. A discrete stochastic process is a *Markov chain* $\mathcal{C} = (C_0, C_1, C_2, ...)$ iff $\forall k \in \mathbb{N}$. $P(C_k|C_{k-1}, C_{k-2}, ..., C_1, C_0) = P(C_k|C_{k-1})$. A Markov chain on a sample space S can also be defined as follows:

▶ **Definition 1.** A tuple $\mathcal{C} = (S, s_0, P)$ is a Markov Chain (MC), if S is a finite set of states, $s_0 \in S$ is the initial state and P is a single $|S| \times |S|$ probability transition matrix, so $\forall s, t \in S$. $P_{s,t} \geq 0$ and $\forall s \in S$. $\sum_{t \in S} P_{s,t} = 1$.

The probability of transitioning from any state s to a state t in k steps can be found as the entry of index (s,t) in P^k [8]. We write $\pi^{(k)}$ for the probability distribution vector over S at time k and $\pi_s^{(k)}$ the probability of visiting the state s at time k; note that $\pi^{(k)} = \pi_0 P^k$, where $\pi_s^{(0)}$ is 1 if $s = s_0$ and 0 otherwise. A probability distribution $\bar{\pi}$ over the states of the chain is *stationary* if $\bar{\pi} = \bar{\pi}P$. Given an initial distribution $\pi^{(0)}$ we compute the unique stationary *limit distribution* μ as $\mu = \lim_{k\to\infty} \pi^{(0)} P^k$.

We write ξ_s for the *expected residence time* of state $s \in S$: $\xi_s = \sum_{k=0}^{\infty} P^k_{s_0,s}$. A state $s \in S$ is *absorbing* if $P_{s,s} = 1$. In the figures we do not draw the looping transition of the absorbing states, to reduce clutter.

We will enrich our Markovian models with a finite set $\mathtt{V}$ of natural-valued variables, and for simplicity we assume that there is a very large finite bit-size M such that a variable is at most M bit long. We define an assignment function $A : S \to [0, 2^M - 1]^{|\mathtt{V}|}$ assigning to each state the values of the variables in that state. We write $\mathtt{v}(s)$ to denote the value of the variable $\mathtt{v} \in \mathtt{V}$ in the state $s \in S$. Consider a stochastic process representing the value of a variable $\mathtt{v}$ over time, derived fro the behavior of a Markov chain labeled with valuations of this variable. We will call this process the *marginal process*, or just *marginal*, $\mathcal{C}_{|\mathtt{v}}$ on $\mathtt{v}$, formally:

▶ **Definition 2.** Let $\mathcal{C} = (S, s_0, P)$ be a Markov chain and $\mathtt{v} \in \mathtt{V}$ a variable. Then we define the *marginal process* $\mathcal{C}_{|\mathtt{v}}$ *of* $\mathcal{C}$ *on* $\mathtt{v}$ as a stochastic process $(\mathtt{v}_1, \mathtt{v}_2, ...)$ where $\forall n.\ P(\mathtt{v}_k = n) = \sum_{\{s|\mathtt{v}(s)=n\}} \pi_s^{(k)}$

We will use $\mathtt{v}$ to denote the marginal process when it is clear from the context that we refer to it. Note that $\mathcal{C}_{|\mathtt{v}}$ is not necessarily a Markov chain. When it is, it can be drawn like in Fig. 3bcd. In the paper we will allow assignments of sets of values to variables and marginals on sets of variables; such extensions are straightforward, since multiple variables can be seen as a single variable on their product space. Assume that the system modeled by $\mathcal{C}$ has a single secret variable $\mathtt{h}$ and a single observable variable $\mathtt{o}$. Then the distributions over the marginal processes $\mathcal{C}_{|\mathtt{h}}$ and $\mathcal{C}_{|\mathtt{o}}$ model the behavior of the secret and observable variable respectively at each time step, and their correlation quantifies the amount of information about the secret that can be inferred by observing the observable variable.

Entropy is a measure of the uncertainty of a probability distribution. The following definitions are standard:

▶ **Definition 3** ([8])**.** Let X and Y be two random variables with probability mass functions $p(x)$ and $p(y)$ respectively and joint probability mass function $p(x,y)$. Then we define the following non-negative real-valued functions:

- Entropy $H(X) = -\sum_{x\in X} p(x) \log_2 p(x)$
- Joint entropy $H(X,Y) = -\sum_{x\in X} \sum_{y\in Y} p(x,y) \log_2 p(x,y)$
- Conditional entropy $H(X|Y) = -\sum_{x\in X} \sum_{y\in Y} p(x,y) \log_2 p(x|y) =$
  $= \sum_{y\in Y} p(y) H(X|Y=y) = -\sum_{y\in Y} p(y) \sum_{x\in X} p(x|y) \log_2 p(x|y) =$
  $= H(X,Y) - H(Y)$ (chain rule)
- Mutual information $I(X;Y) = \sum_{x\in X} \sum_{y\in Y} p(x,y) \log_2 \left(\frac{p(x,y)}{p(x)p(y)}\right) =$
  $= H(X) + H(Y) - H(X,Y) \leq \min(H(X), H(Y))$

Mutual information can be generalized to two vectors of random variables $\bar{X}, \bar{Y}$ as $I(\bar{X};\bar{Y}) = \sum_{\bar{x}\in\bar{X}} \sum_{\bar{y}\in\bar{Y}} p(\bar{x},\bar{y}) \log_2 \left(\frac{p(\bar{x},\bar{y})}{p(\bar{x})p(\bar{y})}\right)$.

▶ **Definition 4.** [8] Let $\mathcal{X} = (X_1, X_2, ...)$ and $\mathcal{Y} = (Y_1, Y_2, ...)$ be two stochastic processes. Then we define the following non-negative real-valued functions:

- Entropy $H(\mathcal{X}) = \lim_{k\to\infty} H(X_1, X_2, ..., X_k)$

- Entropy rate $\bar{H}(\mathcal{X}) = \lim_{k\to\infty} \frac{1}{k} H(X_1, X_2, ..., X_k)$ when the limit exists
- Mutual information $I(\mathcal{X}; \mathcal{Y}) = \lim_{k\to\infty} I(X_1, X_2, ..., X_k; Y_1, Y_2, ..., Y_k)$
- Mutual information rate $\bar{I}(\mathcal{X}; \mathcal{Y}) = \lim_{k\to\infty} \frac{1}{k} I(X_1, X_2, ..., X_k; Y_1, Y_2, ..., Y_k)$ when the limit exists

Entropy and mutual information of stochastic processes always exist, as shown in Section 3. Entropy rate and mutual information rate may not exist in general, but exist when the stochastic processes are Markov chains [8]; we discuss them in Section 4.

Since every state s in a MC $\mathcal{C} = (S, s_0, P)$ has a discrete probability distribution over the successor states we can calculate the entropy of this distribution, the *local entropy*:

▶ **Definition 5.** Let $\mathcal{C} = (S, s_0, P)$ be a Markov chain. Then for each state $s \in S$ we define the *local entropy* of s as $L(s) = -\sum_{t\in S} P_{s,t} \log_2 P_{s,t}$

Note that $L(s) \leq \log_2(|S|)$ [5]. If a stochastic process is a Markov chain $\mathcal{C}$, its entropy $H(\mathcal{C})$ can be computed by considering the local entropy $L(s)$ as the expected reward of a state s and then computing the expected total reward of the chain [5]: $H(\mathcal{C}) = \sum_{s\in S} L(s)\xi_s$. It is also known that the entropy rate can be computed similarly by summing the local entropies of each state weighted by the state's probability in the limiting distribution [8]: $\bar{H}(\mathcal{C}) = \sum_{s\in S} L(s)\mu_s$.

In this paper we use information theory to compute the amount of bits of a secret variable `h` that can be inferred by an attacker able to observe the value of an observable variable `o` at any moment in time. We call this amount *Shannon leakage* or just leakage, and it corresponds to the mutual information $I(\mathcal{C}_{|\mathtt{o}}, \mathcal{C}_{|\mathtt{h}})$ between the marginal on the secret and the marginal on the observable variable.

Operationally, Shannon leakage is related to the number of attempts that an attacker has to do to guess the value of the secret. Other leakage measures exist, but Shannon leakage is the only one for which the chain rule of Definition 3 holds; since the chain rule is used in many results in this work, we do not expect such results to extend to other leakage measures.

```
1 secret int1 h;
2 observable int1 o;
3 public int1 r;
4 random r := randombit
      (0.75);
5 assign o := h ^ r;
6 return;
```

Figure 2 Bit XOR example: source code.

The modeling of a process as a Markov chain in our context starts by dividing the variables in private and public variables. Private variables, including the secret variable `h`, are the ones whose value is not defined at compilation time. In each state of the Markov chain a set of allowed values is assigned to each private variable. Public variables, including the observable variable `o` and the program counter `pc`, are variables whose value is known to the analyst. On each state a given value is assigned to each public variable.

Given the source code and a prior distribution over the private variables, we have enough information to build a Markov chain representing the semantics, since for each state we can determine its successor states and the corresponding transition probabilities. We show a simple example, and refer to [4] for the complete semantics. The source code for the example is shown in Fig. 2 and the corresponding Markov chain semantics in Fig. 3a.

Let `h` be a secret bit, `o` an observable bit and `r` a random bit being assigned the value 0 with probability 0.75 and 1 otherwise. We assign to `o` the result of the exclusive OR between `h` and `r` and terminate. We want to quantify the amount of information about `h` that can be inferred by knowing the value of `o`.

To compute the leakage we need to compute three marginals from the Markov chain semantics:

Joint marginal The joint marginal process $\mathcal{C}_{|(o,h)}$ models the joint behavior of the secret and observable variables. It is shown in Fig. 3b.

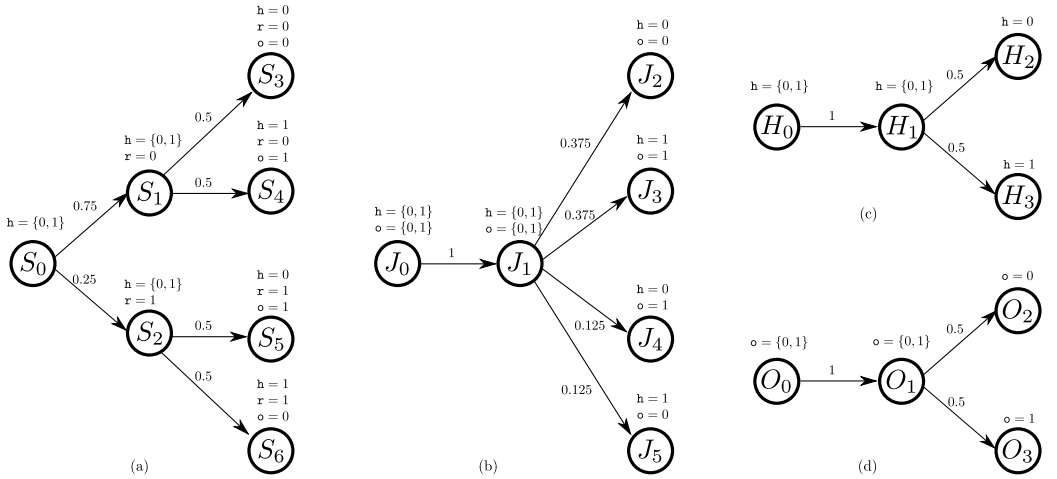

Figure 3 Bit XOR example: a) Markov chain semantics $\mathcal{C}$. b) Joint marginal $\mathcal{C}_{|(o,h)}$. c) Secret's marginal $\mathcal{C}_{|h}$. d) Observer's marginal $\mathcal{C}_{|o}$.

Secret's marginal The secret's marginal process $\mathcal{C}_{|h}$ models the behavior of the secret variable. It is shown in Fig. 3c.

Observer's marginal The observer's marginal process $\mathcal{C}_{|o}$ models the behavior of the observable variable. It is shown in Fig. 3d.

Finally we compute the mutual information between the secret and observable variable using the formula $I(X;Y) = H(X) + H(Y) - H(X,Y)$, obtaining $I(\mathtt{o};\mathtt{h}) = I(\mathcal{C}_{|o};\mathcal{C}_{|h}) = H(\mathcal{C}_{|o}) + H(\mathcal{C}_{|h})) - H(\mathcal{C}_{|(o,h)})) = 1 + 1 - 1.8112... \approx 0.1887$ bits, proving that the program leaks ≈ 0.1887 bits, or 18.87% of the secret.

3 Non-terminating Processes with Finite Leakage

The Markov chain semantics of the system describes the joint behavior of all variables. To compute information leakage we are only interested in the secret and the observable variables, so we can restrict to them only for simplicity. We assume that the system has a single secret variable $\mathtt{h}$ with uniform prior distribution and a single observable variable $\mathtt{o}$, but the procedure does not change for multiple secret or observable variables. We remark that, even though the attacker can perform multiple observations, we do not model the case in which the attacker actually interacts with the system. In such case directed information would have to be used as the leakage metric instead of mutual information. We refer to Alvim et al. [2] for the details.

The behavior of the secret variable $\mathtt{h}$ is modeled by the marginal $\mathcal{C}_{|\mathtt{h}}$, and similarly the behavior of $\mathtt{o}$ is modeled by $\mathcal{C}_{|\mathtt{o}}$ and the joint behavior of the two variables by $\mathcal{C}_{|\mathtt{o},\mathtt{h}}$. The following lemma shows the existence of the entropy values of such marginals and a sufficient condition for the finiteness of their mutual information:

▶ **Theorem 6.** *Let $\mathcal{C} = (S, s_0, P)$ be a Markov chain with secrets and observations and $\mathcal{C}_{|\mathtt{o}}$ and $\mathcal{C}_{|\mathtt{h}}$ its marginals on the observable and secret variables, respectively. Then:*

- *$H(\mathcal{C}_{|\mathtt{o}})$, $H(\mathcal{C}_{|\mathtt{h}})$ and $I(\mathcal{C}_{|\mathtt{o}};\mathcal{C}_{|\mathtt{h}})$ exist;*
- *$H(\mathcal{C}_{|\mathtt{o}}) < \infty \vee H(\mathcal{C}_{|\mathtt{h}}) < \infty \Rightarrow I(\mathcal{C}_{|\mathtt{o}}, \mathcal{C}_{|\mathtt{h}}) < \infty$*

Intuitively, if $H(\mathcal{C}_{|\mathtt{o}}) < \infty$ and $H(\mathcal{C}_{|\mathtt{h}}) = \infty$ then there is an infinite number of secret bits but only a finite amount of observations we can analyze. In the opposite case where $H(\mathcal{C}_{|\mathtt{o}}) = \infty$ and $H(\mathcal{C}_{|\mathtt{h}}) < \infty$ we can analyze an infinite number of observations, but there is only a finite amount of secret bits to be discovered.

```
1  secret int1 h;
2  if (h==0) then
3    observable int1 o:=0;
4    while (0==0) do
5      random o:=randombit
           (0.5);
6    od
7  else
8    observable int1 o:=0;
9    while (0==0) do
10     random o:=randombit
           (0.75);
11   od
12 fi
13 return;
```

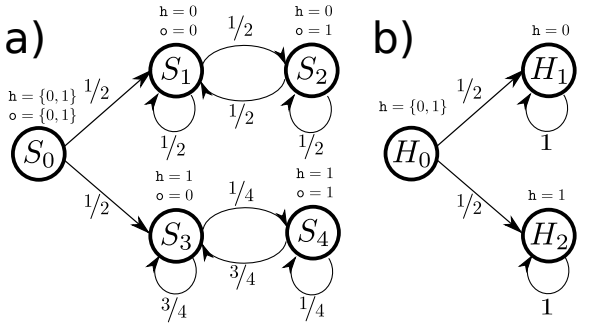

Figure 4 Non-terminating leaking program example. On the left: program code. On the right: Markov chain semantics a) joint marginal $\mathcal{C}_{|o,h}$ b) secret's marginal $\mathcal{C}_{|h}$

If either $H(\mathcal{C}_{|\mathtt{o}}) = \infty$ or $H(\mathcal{C}_{|\mathtt{h}}) = \infty$ then $H(\mathcal{C}_{|\mathtt{o},\mathtt{h}}) = \infty$, since $H(X,Y) \geq \max(H(X), H(Y))$. It follows that if either the observation or the secret are infinite but not both, the formula $I(\mathcal{C}_{|\mathtt{o}}, \mathcal{C}_{|\mathtt{h}}) = H(\mathcal{C}_{|\mathtt{o}}) + H(\mathcal{C}_{|\mathtt{h}}) - H(\mathcal{C}_{|\mathtt{o},\mathtt{h}})$ will produce an indeterminate form $\infty - \infty$ and thus cannot be directly used to compute the leakage. Nonetheless, in both cases leakage has a finite value by Theorem 6.

If any marginal is a Markov chain, it is possible to compute its entropy in polynomial time in the size of the chain [5]. Otherwise, consider that the entropies of the marginals are limit computations, since $H(\mathcal{C}_{|\mathtt{v}}) = \lim_{k\to\infty} H(\mathtt{v}_1, ..., \mathtt{v}_k)$. This allows us to compute mutual information using the limit of the entropies of the marginal processes:

$$
\begin{aligned}
I(\mathcal{C}_{|\mathtt{o}}; \mathcal{C}_{|\mathtt{h}}) &= \lim_{k\to\infty} I(\mathtt{o}_1, ..., \mathtt{o}_k; \mathtt{h}_1, ..., \mathtt{h}_k) \\
&= \lim_{k\to\infty} \left(H(\mathtt{o}_1, ..., \mathtt{o}_k) + H(\mathtt{h}_1, ..., \mathtt{h}_k) - H((\mathtt{o},\mathtt{h})_1, ..., (\mathtt{o},\mathtt{h})_k)\right)
\end{aligned}
$$

The limit above computes information leakage in any case, but is not always the most efficient option available. When it is known that the secret (resp. observation) is finite, it is more efficient to use the formula $I(\mathcal{C}_{|\mathtt{o}}; \mathcal{C}_{|\mathtt{h}}) = H(\mathcal{C}_{|\mathtt{h}}) - H(\mathcal{C}_{|\mathtt{h}}|\mathcal{C}_{|\mathtt{o}})$ (resp. $I(\mathcal{C}_{|\mathtt{o}}; \mathcal{C}_{|\mathtt{h}}) = H(\mathcal{C}_{|\mathtt{o}}) - H(\mathcal{C}_{|\mathtt{o}}|\mathcal{C}_{|\mathtt{h}})$). Remember that when both secret and observation are finite the process terminates, so the procedure we proposed in [4] can be used with some additional assumptions.

3.1 Example: A Non-terminating Program on a Finite Secret

We now solve a case in which the secret is finite and Markovian and the observation infinite. Consider a program (Figure 4) with a secret bit h. If h is 0 the program produces an infinite string of zeroes and ones with the same probability 0.5, starting with a zero. If h is 1 the program also produces a string of zeroes and ones starting with a zero, but the probability that it will produce a zero is 0.75. Note that this program cannot be encoded as a finite channel matrix, as it has an infinite amount of possible outputs.

An attacker may be able to observe this infinite string and infer information about the secret by studying the frequencies of zeroes and ones. The attacker starts with no knowledge of the secret, which is encoded as an initial uniform distribution over the secret bit h. Reasonably, an attacker observing the output for an infinite time would be able to decide whether the frequency of zeroes is 0.5 or 0.75 and infer the value of h consequently. The Markov chain semantics for it is shown in Fig. 4a on the right.

Since $\mathtt{h}_1 = \mathtt{h}_2 = \mathtt{h}_3 = ...$ we will just call it $\mathtt{h}$. The behavior of $\mathtt{h}$ is modeled by the Markov chain in Fig. 4b on the right, and its entropy is $H(\mathtt{h}) = 1$ bit. $H(\mathcal{C}_{|\mathtt{h}}|\mathcal{C}_{|\mathtt{o}})$ corresponds to $\lim_{k\to\infty} H(\mathtt{h}|\mathtt{o}_1, ..., \mathtt{o}_k)$.

We compute $H(\mathtt{h}|\mathtt{o}_1, ..., \mathtt{o}_k)$ for $k \to \infty$. Note that at time 1 $\mathtt{o}$ is always 0, then it changes randomly depending on the value of $\mathtt{h}$. We will write down the joint distribution of $\mathtt{h}$ and $\mathtt{o}$ as a function of k and use it to compute the marginal over $\mathtt{o}$ and finally the conditional entropy.

The joint distribution of $\mathtt{h}$ and $\mathtt{o}$ is shown in the Appendix due to space constraints. Now let $w^k \in \{0,1\}^k$ be a sequence of k bits. Consider the formula for conditional entropy:

$$H(\mathtt{h}|\mathtt{o}_1, ..., \mathtt{o}_k) = \sum_{w^k \in \{0,1\}^k} P(\mathtt{o}_1, ..., \mathtt{o}_k = w^k) H(\mathtt{h}|\mathtt{o}_1, ..., \mathtt{o}_k = w^k) \tag{1}$$

In our case it holds that $H(\mathtt{h}|\mathtt{o}) = \lim_{k\to\infty} H(\mathtt{h}|\mathtt{o}_1, ..., \mathtt{o}_k) = 0$ thus $I(\mathtt{o};\mathtt{h}) = H(\mathtt{h}) - H(\mathtt{h}|\mathtt{o}) = 1 - 0 = 1$ bit. The leakage of the program in Fig. 4 is 1 bit, proving that an attacker able to analyze the bit streams produced by the system will eventually learn the value of the secret $\mathtt{h}$ with an arbitrary confidence. Note that this considers an attacker able to observe the system for an infinite time.

More importantly, note that the marginal process of $\mathtt{o}$ is *not* a Markov chain. This depends on the fact that the joint distribution depends also on the information that the attacker has about $\mathtt{h}$, so while the attacker gathers information about $\mathtt{o}$ and $\mathtt{h}$ the joint distribution changes and thus the marginal distribution of $\mathtt{o}$ changes also. Nonetheless, the marginal process can be represented in a closed form like the one in Fig. 6d.

4 Leakage Rate of a Markov Chain

In the case in which $H(\mathcal{C}_{|\mathtt{o}}) = \infty$ and $H(\mathcal{C}_{|h}) = \infty$, i.e. when the secret is an infinite number of bits and the observer can observe it for an infinite time, then the leakage $I(\mathtt{o}, \mathtt{h})$ can be infinite. In this case it is more interesting to compute how much information the process leaks for each time step. This quantity is known as *leakage rate*, and corresponds to the *mutual information rate* of the secret and observable.

Note that the computation of leakage as a rate over time assumes that the attacker is able to keep track of the discrete time, so in this section we will assume that every constant-time operation takes 1 time step. This can be equivalently stated as saying that all transitions between states of the Markov chain semantics represent observable steps.

To compute leakage rate, we encode the process-attacker scenario with a Markov chain as shown in Section 2 and compute the joint, secret and attacker's marginal, which may not be Markovian. Then we can use the marginals to compute leakage rate by applying the following definition:

▶ **Definition 7.** Let $\mathcal{C} = (S, s_0, P)$ be a Markov chain and $\mathcal{C}_{|\mathtt{o},\mathtt{h}}$, $\mathcal{C}_{|\mathtt{o}}$ and $\mathcal{C}_h$ its marginals on $(\mathtt{o},\mathtt{h})$, $\mathtt{o}$ and $\mathtt{h}$ respectively. Then the *leakage rate* $\bar{I}$ is defined as $\bar{I}(\mathcal{C}_{|\mathtt{o}}; \mathcal{C}_h) = \bar{H}(\mathcal{C}_{|\mathtt{o}}) + \bar{H}(\mathcal{C}_{|\mathtt{h}}) - \bar{H}(\mathcal{C}_{|\mathtt{o},\mathtt{h}})$.

Leakage rate can also be computed as a limit, since

$$\begin{aligned}\bar{I}(\mathcal{C}_{|\mathtt{o}}; \mathcal{C}_{|\mathtt{h}}) &= \lim_{k\to\infty} \frac{I(\mathtt{o}_1, ..., \mathtt{o}_k; \mathtt{h}_1, ..., \mathtt{h}_k)}{k} \\ &= \lim_{k\to\infty} \frac{(H(\mathtt{o}_1, ..., \mathtt{o}_k) + H(\mathtt{h}_1, ..., \mathtt{h}_k) - H((\mathtt{o},\mathtt{h})_1, ..., (\mathtt{o},\mathtt{h})_k))}{k}\end{aligned}$$

when the limit exists.

Generally both entropy and leakage rate could be infinite, for instance for a program that leaks 1 bit at time 1, 2 bits at time 2, and so on, the leakage rate would be infinite. Since we postulated that there exists a very large but finite maximum size M for the variables declared in the system,

Data: A Markov Chain $\mathcal{C} = (S, s_0, P)$ and its initial probability distribution $\pi^{(0)}$.
Result: The limit probability distribution μ of the chain.

foreach *transient state t* **do**
  $\mu_t \leftarrow 0$;
  foreach $u \in S \setminus \{t\}$ **do**
    $\pi_u^{(0)} \leftarrow \pi_u^{(0)} + \pi_t^{(0)} \frac{P_{t,u}}{1-P_{t,t}}$
    foreach $s \in S \setminus \{t\}$ **do**
      $P_{s,u} \leftarrow P_{s,u} + P_{s,t} \frac{P_{t,u}}{1-P_{t,t}}$
      $P_{s,t} \leftarrow 0$
    end
  end
end
foreach *end component* R_i **do**
  Let $\pi_{R_i}^{(\infty)} = \sum_{r \in R_i} \pi_s^{(0)}$ and E_i be a system of linear equations;
  foreach $r \in R_i$ **do** Add to E_i the equation $\mu_r = \sum_{r' \in R_i} \mu_{r'} P_{r',r}$;
  Solve the system E' under the condition $\sum_{r \in R_i} \mu_r = \pi_{R_i}^{(\infty)}$ to obtain μ_r for each $r \in R_i$;
end

Algorithm 1: Compute the limit distribution of a Markov chain.

it is impossible to declare an unbounded amount of secret or observable bits on each step of the program execution. We do not think that this restriction limits significantly the programs that can be analyzed, while guaranteeing that the entropy and leakage rate do not diverge to positive infinity is a significantly useful result.

Both entropy rate and leakage rate may still oscillate, even though since they are defined in terms of Cesàro limits this happens only in pathological cases. We do not expect these cases to be common in normal secret-dependent systems, and leave finding a meaningful measure of leakage for these cases an open problem. This reflects similar issues in related definitions of leakage rate [7, 12].

A case in which both entropy and leakage rate exist is when the marginal processes modeling the behavior of the observable and secret variables are both Markovian. Intuitively, this happens when the secret gets periodically replaced with a new one, and thus the information the attacker has on it is reset to the prior information. We will show this with an example in Section 4.1.

When any of the marginals is a Markov chain it is possible to compute its entropy rate efficiently as $\bar{H}(\mathcal{C}) = \sum_{s \in S} L(s)\mu_s$, where μ is the limit distribution of the chain. The entropy rate of a Markov chain with a given initial probability distribution $\pi^{(0)}$ exists and is unique [8].

Computing the limit distribution can be accomplished on irreducible Markov chains by solving a system of linear equations, but the Markov chains we consider are usually reducible, so a new algorithm is required. Algorithm 1 computes the limit distribution of any Markov Chain $\mathcal{C} = (S, s_0, P)$. The algorithm uses the well-established concept of end components of a MC; each end component R_i behaves as an irreducible MC, so it is sufficient to compute the probability $\pi_{R_i}^{(\infty)}$ of eventually visiting R_i and redistribute $\pi_{R_i}^{(\infty)}$ among the states of R_i by solving a system of linear equations.

▶ **Theorem 8.** *Algorithm 1 terminates in polynomial time in $|S|$ and when it does it returns the limit distribution μ.*

Due to space constraints we refer to the Appendix for the proof of this theorem and a full explanation of the steps of Algorithm 1.

Since computing the local entropy of each state in time $O(|S|^2)$ is trivial, the formula $\bar{H}(\mathcal{C}) = \sum_{s \in S} L(s)\mu_s$ can be used to compute entropy rate of a Markov chain in polynomial time. Note that this is a particular case of the computation of an expected infinite-horizon reward rate of a reward function defined on the transitions of a Markov chain.

Having computed the entropy rates of the joint, secret and attacker's marginals we can apply Definition 7 to obtain the leakage rate of the system.

4.1 Example: Leaking Mix Node Implementation

We show an example of a program leaking an infinite amount of information and we compute its leakage rate using the method described above.

A mix node [9] is a program meant to scramble the order in which packages are routed through a network, to increase the anonymity of the sender. Even if the packages are encrypted, some information about the sender could be inferred by observing the order in which they are forwarded. A mix node changes this order to a random one, thus making it harder for an attacker to connect each package to its sender.

```
1  secret int3 inorder;
2  public int3 rand;
3  observable int3 outorder;
4  while (0==0) do
5    assign inorder := [0,5];
6    random rand := random
         (0,5);
7    assign outorder :=
8        (inorder ^ rand)%6;
9  od
10 return;
```

Figure 5 A leaking implementation of a mix node.

A mix node waits until it has accumulated a fixed amount of packages, and then forwards them in a random order. If the exit order of the packages is independent from the entrance order, then no information about the latter can be inferred by observing the former. We will present an implementation of a mix node where the entrance and exit order are not independent and compute the rate of the information leakage.

The implementation of the mix node is shown in Fig. 5. This particular node waits until it has accumulated 3 packages and then sends them in a random order. Naming the packages A, B and C there are 6 possible entrance orders: ABC, ACB, BAC, BCA, CAB, and CBA. We will number them from 0 to 5.

In line 5 of the code a random number from 0 to 5 is assigned to the secret variable `inorder`, modeling the secret entrance order. Then in line 6 a random value uniformly distributed from 0 to 5 is assigned to the variable `rand`. Finally, in line 7 the bitwise exclusive OR modulo 6 of the variables `inorder` and `rand` is assigned to the observable variable `outorder`, which represents the order in which the packages exit from the mix node and is observable to the attacker. After producing an exit order the mix node receives three more packages in a new entrance order, scrambles them the same way and forwards them, and so on forever.

Assume that the prior distribution over the input order is uniform. The resulting probability distribution on the exit order is $P(\texttt{outorder}) = \{0 \mapsto {}^5\!/_{18}, 1 \mapsto {}^5\!/_{18}, 2 \mapsto {}^2\!/_{18}, 3 \mapsto {}^2\!/_{18}, 4 \mapsto {}^2\!/_{18}, 5 \mapsto {}^2\!/_{18}\}$. This depends on the fact that bitwise OR and modulo operations do not preserve distribution uniformity.

The Markov chain semantics of the system has more than 300 states. It can be computed and analyzed automatically in less than a second. Assuming that each line of code is executed in one time step, the entropy rates of the marginals are $\bar{H}(\texttt{inorder}) = 0.86165...$ bits, $\bar{H}(\texttt{outorder}) = 0.82766...$ and $\bar{H}(\texttt{inorder}, \texttt{outorder}) = 1.59367...$, giving a leakage rate of $\bar{I}(\texttt{inorder}; \texttt{outorder}) = \bar{H}(\texttt{inorder}) + \bar{H}(\texttt{outorder}) - \bar{H}(\texttt{inorder}, \texttt{outorder}) = 0.86165... + 0.82766... - 1.59367... \approx 0.09564$ bits.

The leakage rate for each time unit is ≈ 0.09564 bits. Since the entropy rate of the secret is ≈ 0.86165 bits, we can conclude that this implementation of a mix node has a rate of leakage

Data: A Markov Chain $\mathcal{C} = (S, s_0, P)$ with the variables $\mathtt{o}$ and $\mathtt{h}$, two integers t_1 and t_2 satisfying $t_1 \leq t_2$.
Result: The leakage from time t_1 to time t_2 $I^{(t_1,t_2)}(\mathtt{o},\mathtt{h})$

for $x \in \{\mathtt{o}, \mathtt{h}, (\mathtt{o},\mathtt{h})\}$ **do**
  Compute the marginal $\mathcal{C}_{|x}$, and let $\pi^{(t_1)}_{|x}$ be the probability distribution over its states at time t_1 and $H^{(t_1)}(\mathcal{C}_{|x}) = H(\pi^{(t_1)}_{|x})$;
end
Compute $I^{(t_1,t_1)} = H^{(t_1)}(\mathcal{C}_{|\mathtt{o}}) + H^{(t_1)}(\mathcal{C}_{|\mathtt{h}}) - H^{(t_1)}(\mathcal{C}_{|\mathtt{o},\mathtt{h}})$;
for $i = t_1 + 1$ *to* t_2 **do**
  for $x \in \{\mathtt{o}, \mathtt{h}, (\mathtt{o},\mathtt{h})\}$ **do**
    $H^{(i)}(\mathcal{C}_{|x}) \leftarrow H^{(i-1)}(\mathcal{C}_{|x}) + \sum_{s \in S_{|x}} \pi^{(i-1)}_{|x}(s) L_{|x}(s)$;
    $\pi^{(i)}_{|x} \leftarrow \pi^{(i-1)}_{|x} P^{(i-1,i)}_{|x}$;
  end
  $I^{(t_1,i)} \leftarrow H^{(i)}(\mathcal{C}_{|\mathtt{o}}) + H^{(i)}(\mathcal{C}_{|\mathtt{h}}) - H^{(i)}(\mathcal{C}_{|\mathtt{o},\mathtt{h}})$;
end
return $I^{(t_1,t_2)}$;

Algorithm 2: Compute the leakage of a MC from a time t_1 to a time t_2.

$0.09564/0.86165 \approx 11.1\%$ of each of its infinite secrets.

Note that in this simple case the loop always takes 3 time units to complete, so it would have been possible to just compute the leakage of one loop and divide it by 3, but in general loops do not compute for a fixed number of time units, e.g. if they contain multiple `return` statements.

5 Bounded Time/Leakage Analysis

We consider two similar bounded approaches to the leakage problem: computing the leakage of a Markov chain within a given time frame, or computing how long it takes for the Markov chain to leak a given amount of information.

5.1 Bounded Time

We want to compute the leakage for an attacker that is able to observe the behavior of the program for $t < \infty$ time units. We abstract time by considering each time unit as a step in the evolution of the Markov chain modeling the system. The definition of mutual information from a time t_1 to a time $t_2 > t_1$ is as follows:

▶ **Definition 9.** Let $\mathcal{X}$ and $\mathcal{Y}$ be two stochastic processes. Then the mutual information between X_i and Y_i from time t_1 to time t_2 is $I(X_{t_1}, ..., X_{t_2}; Y_{t_1}, ..., Y_{t_2}) = H(X_{t_1}, ..., X_{t_2}) + H(Y_{t_1}, ..., Y_{t_2}) - H(X_{t_1}, ..., X_{t_2}, Y_{t_1}, ..., Y_{t_2})$

We will refer to it as $I^{(t_1,t_2)}(\mathcal{X};\mathcal{Y})$ for simplicity. Consider as usual the Markov chain semantics $\mathcal{C} = (S, s_o, P)$ modeling the behavior of the system. We present an iterative algorithm to compute $I^{(t_1,t_2)}(\mathtt{o},\mathtt{h})$ in time $O(t_2|S|^2)$: the algorithm first computes the distribution at time t_1 and then computes the behavior of the chain until time t_2 while keeping track of the amount of leakage accumulated. In the algorithm let $S_{|x}$ be the state space of the marginal, $\pi^{(i)}_{|x}$ the distribution on the marginal at time i, and $P^{(i,j)}_{|x}$ the probability of transitioning from i to j in the marginal.

▶ **Theorem 10.** *Algorithm 2 terminates in time $O(t_2|S|^2)$ and when it does it outputs $I^{(t_1,t_2)}(o,h)$.*

Note that Algorithm 2 is pseudopolynomial, as it depends not only on the size of the chain but also on the parameter t_2. Also note that due to the Markov property it holds that $I^{(t_1,t_2)} = I^{(t'_1,t'_2)}$ whenever $\pi^{(t_1)}_{|o,h} = \pi^{(t'_1)}_{|o,h}$ and $t_2 - t_1 = t'_2 - t'_1$.

5.2 Bounded Leakage

We want to determine how many time units it takes for the system to leak a given amount c of bits of information. The problem is more complex than the one analyzed in the previous section, since leakage is a complex function of the behavior of the system in time and finding a way to bound or reverse it is not obvious.

We start by considering the qualitative version of the problem: does there exist a time t such that $I_t(\mathcal{C}_O, \mathcal{C}_h) \geq c$? To answer we note that the the sequence of leakages is monotonic non-decreasing over time, so if the answer is yes then the leakage will remain greater than c for each time $t' \geq t$. This allows us to answer the qualitative question by computing the leakage on the infinite time horizon as shown in Section 3; let it be l^∞. If $l^\infty < c$ then there is no time t such that the leakage is c, while if $l^\infty > c$ then such time exists. If $l^\infty = c$ then the system leaks c bits on the infinite time horizon but we have no guarantee that this amount will be reached in finite time.

In the case in which $l^\infty \geq c$ we can ask the quantitative question, i.e. at what time t the system will have leaked at least c bits. If $l^\infty > c$ we know that such time t exists, while if $l^\infty = c$ it may not. We will define the *bounded leakage problem* as follows: given a Markov chain $\mathcal{C} = (S, s_0, P)$ labeled with secrets and observations and a positive real number c, determine if there exists a finite time t such that the information leakage of the chain at time t is exactly c.

The problem is harder than it seems. For deterministic programs, it has been shown by Terauchi that it is not a k-safety property for any k [17]. The problem has also been addressed computationally by Heusser and Malacaria [10]. For randomized programs, we will show that the problem can be reduced from Skolem's problem [14]. While smaller instances have been shown to be decidable, the full decidability of Skolem's problem is still an open question [15]. Akshay et al. [1] show that Skolem's problem is equivalent to the following: given a Markov chain $\mathcal{C} = (S, s_0, P)$, a state s and a probability r determine whether there is a time t such that $\pi^{(t)}_s = r$. We will call this *Skolem's Markov chain reachability problem*. Intuitively, information leakage is a harder problem than reachability, as formally stated by the following theorem:

▶ **Theorem 11.** *Let $\mathbb{A}$ be an algorithm deciding the bounded leakage problem. Then $\mathbb{A}$ decides Skolem's Markov chain reachability problem.*

6 Conclusions and Related Work

We have shown how to provide meaningful measures of the effectiveness of a secret-dependent non-terminating program in protecting its secret, by computing Shannon leakage when its value is finite and Shannon leakage rate otherwise. Operationally, Shannon leakage is related to the expected number of guesses it will take for the attacker to find out the secret's value, so leakage and leakage rate can be used to understand the amount of time that the attacker will require to infer the system's secret [13]. To the same aim, we provided an algorithm that computes the amount of leakage from a program in a given time frame. Finally, we have shown that a precise quantification of the time required to leak a given amount is a hard problem, proving the complexity of the problem.

The quantification of leakage for an infinite observation and a finite or infinite secret has been recently considered by Chothia et al. [7]. Their framework is different from ours as they study

probabilistic "point to point" leakage; also they provide no algorithms to compute leakage. The authors do not consider the case in which the observation is finite and the secret infinite. Also, in the infinite secret and observation case they explicitly do not consider the leakage rate per time unit, preferring to compute the leakage for each occurrence of the `secret` command in their framework.

Alvim et al [2] study the setting of interactive systems, where secrets and observables can alternate during the computation and influence each other. They show that in this case mutual information is only an upper bound on leakage and "direct information" is a more precise leakage measure. This work is related to ours in that it investigates multi-stage processes but it presents significant differences as it doesn't investigate infinite leakage nor Markovian processes.

Recently Backes et al. also present a method for leakage rate computation based on stationary distribution of Markov chains, which they compute using PageRank [3]. We expect that Algorithm 1 would be a useful addition to their approach.

References

1 S. Akshay, J. Ouaknine, T. Antonopoulos, and J. Worrell. Reachability problems for Markov chains. Personal communication, November 2013.

2 M. S. Alvim, M. E. Andrés, and C. Palamidessi. Quantitative information flow in interactive systems. *Journal of Computer Security*, 20(1):3–50, 2012.

3 M. Backes, G. Doychev, and B. Köpf. Preventing side-channel leaks in web traffic: A formal approach. In *NDSS*. The Internet Society, 2013.

4 F. Biondi, A. Legay, P. Malacaria, and A. Wasowski. Quantifying information leakage of randomized protocols. In R. Giacobazzi, J. Berdine, and I. Mastroeni, editors, *VMCAI*, 2013.

5 F. Biondi, A. Legay, B. F. Nielsen, and A. Wasowski. Maximizing entropy over Markov processes. In A. H. Dediu and C. Martín-Vide, editors, *LATA*, 2013.

6 K. Chatzikokolakis, C. Palamidessi, and P. Panangaden. Anonymity protocols as noisy channels. *Inf. Comput.*, 206(2-4):378–401, 2008.

7 T. Chothia, Y. Kawamoto, C. Novakovic, and D. Parker. Probabilistic point-to-point information leakage. In *CSF*, pages 193–205. IEEE, 2013.

8 T. Cover and J. Thomas. *Elements of Information Theory*. Wiley, 2012.

9 G. Danezis, R. Dingledine, and N. Mathewson. Mixminion: Design of a type iii anonymous remailer protocol. In *IEEE Symposium on Security and Privacy*, pages 2–15. IEEE, 2003.

10 J. Heusser and P. Malacaria. Quantifying information leaks in software. In C. Gates, M. Franz, and J. P. McDermott, editors, *ACSAC*, pages 261–269. ACM, 2010.

11 B. Köpf, L. Mauborgne, and M. Ochoa. Automatic quantification of cache side-channels. In P. Madhusudan and S. A. Seshia, editors, *CAV*, pages 564–580. Springer, 2012.

12 P. Malacaria. Assessing security threats of looping constructs. In M. Hofmann and M. Felleisen, editors, *POPL*, pages 225–235. ACM, 2007.

13 J. Massey. Guessing and entropy. In *Information Theory, 1994. Proceedings., 1994 IEEE International Symposium on*, pages 204–, Jun 1994.

14 J. Ouaknine. Decision problems for linear recurrence sequences. In L. Gasieniec and F. Wolter, editors, *FCT*, volume 8070 of *Lecture Notes in Computer Science*, page 2. Springer, 2013.

15 J. Ouaknine and J. Worrell. Positivity problems for low-order linear recurrence sequences. In C. Chekuri, editor, *SODA*, pages 366–379. SIAM, 2014.

16 G. Smith. On the foundations of quantitative information flow. In L. de Alfaro, editor, *FOSSACS*, volume 5504 of *Lecture Notes in Computer Science*, pages 288–302. Springer, 2009.

17 H. Yasuoka and T. Terauchi. On bounding problems of quantitative information flow. *Journal of Computer Security*, 19(6):1029–1082, 2011.

Appendix

Fig. 6 depicts the joint distribution of variables $\mathtt{o}$ and $\mathtt{h}$ for the Markov chain semantics in Fig. 4 . For compactness we do not represent the cases with probability 0 in Fig. 6cd, i.e. all the cases in which $\mathtt{o}_1 = 1$.

a)

		$\mathtt{o}_1$	
		0	**1**
$\mathtt{h}$	**0**	$1/2$	0
	1	$1/2$	0

b)

		$\mathtt{o}_1\mathtt{o}_2$			
		00	**01**	**10**	**11**
$\mathtt{h}$	**0**	$1/4$	$1/4$	0	0
	1	$3/8$	$1/8$	0	0

c)

		$\mathtt{o}_1\mathtt{o}_2\mathtt{o}_3$			
		000	**001**	**010**	**011**
$\mathtt{h}$	**0**	$1/8$	$1/8$	$1/8$	$1/8$
	1	$9/32$	$3/32$	$3/32$	$1/32$

d)

		$\mathtt{o}_1\mathtt{o}_2\ldots\mathtt{o}_{k-1}\mathtt{o}_k$					
		$\overbrace{\mathbf{00..00}}^{k}$	$\overbrace{\mathbf{00..01}}^{k}$	$\overbrace{\mathbf{00..10}}^{k}$	$\cdots$	$\overbrace{\mathbf{01..10}}^{k}$	$\overbrace{\mathbf{01..11}}^{k}$
$\mathtt{h}$	**0**	$1/2^k$	$1/2^k$	$1/2^k$	$\cdots$	$1/2^k$	$1/2^k$
	1	$\frac{(3/4)^{k-1}}{2}$	$\frac{(3/4)^{k-2}\cdot 1/4}{2}$	$\frac{(3/4)^{k-2}\cdot 1/4}{2}$	$\cdots$	$\frac{3/4\cdot(1/4)^{k-2}}{2}$	$\frac{(1/4)^{k-1}}{2}$

Figure 6 Non-terminating example: joint distribution of the secret $\mathtt{h}$ and observables $\mathtt{o}_k$: a) for 1 step; b) for 2 steps; c) for 3 steps; d) for k steps.

Proof of Theorem 6. For any variable $\mathtt{v}\in\mathbb{V}$, $H(\mathcal{C}_{|\mathtt{v}})$ exists since $H(\mathcal{C}_{|\mathtt{v}}) = \lim_{k\to\infty} H(\mathtt{v}_1, \mathtt{v}_2, ..., \mathtt{v}_k)$ and $H(\mathtt{v}_1, \mathtt{v}_2, ..., \mathtt{v}_k)$ is a monotonic non-decreasing non-negative sequence on k. Similarly, $H(I(\mathcal{C}_{|\mathtt{o}}; \mathcal{C}_{|\mathtt{h}}))$ exists since also $I(\mathcal{C}_{|\mathtt{o}}; \mathcal{C}_{|\mathtt{h}}) = \lim_{k\to\infty} I(\mathtt{o}_1, \mathtt{o}_2, ..., \mathtt{o}_k; \mathtt{h}_1, \mathtt{h}_2, ..., \mathtt{h}_k)$ and $I(\mathtt{o}_1, \mathtt{o}_2, ..., \mathtt{o}_k; \mathtt{h}_1, \mathtt{h}_2, ..., \mathtt{h}_k)$ is a monotonic non-decreasing non-negative sequence on k.

Now, if any of the two sequences $H(\mathtt{o}_1, \mathtt{o}_2, ..., \mathtt{o}_k)$ and $H(\mathtt{h}_1, \mathtt{h}_2, ..., \mathtt{h}_k)$ converge, then the sequence $\min(H(\mathtt{o}_1, \mathtt{o}_2, ..., \mathtt{o}_k), H(\mathtt{h}_1, \mathtt{h}_2, ..., \mathtt{h}_k))$ is also a convergent monotonic non-decreasing non-negative sequence on k. Since $\forall k \in \mathbb{N}$.

$$I(\mathtt{o}_1, \mathtt{o}_2, ..., \mathtt{o}_k; \mathtt{h}_1, \mathtt{h}_2, ..., \mathtt{h}_k) \leq \min(H(\mathtt{o}_1, \mathtt{o}_2, ..., \mathtt{o}_k), H(\mathtt{h}_1, \mathtt{h}_2, ..., \mathtt{h}_k)) \tag{2}$$

then the sequence $I(\mathtt{o}_1, \mathtt{o}_2, ..., \mathtt{o}_k; \mathtt{h}_1, \mathtt{h}_2, ..., \mathtt{h}_k)$ converges also. ◀

Proof of Theorem 8. Let $\mathcal{C} = \{S, s_0, P\}$ be a Markov chain and $\pi^{(0)}$ its initial distribution. We want to compute its limiting distribution μ.

We call a state t *reachable* from a state s if $\exists k. P^k_{s,t} > 0$. We write $s \leftrightarrow t$ if s is reachable from t and vice versa. A subset of states $R \subseteq S$ is a *end component* if for each pair of states $s, t \in R$ it holds that $s \leftrightarrow t$ and no state in the component has transitions with nonzero probability to states not in the component, i.e. $\sum_{r\in R} P_{r,t} = 0$ for $t \notin R$.

We call a state s *recurrent* if $\xi_s = \infty$, *transient* otherwise. It is known that the states in the end components are all and only the recurrent states of the chain. Thus, the limit probability distribution

Data: A Markov Chain $\mathcal{C} = (S, s_0, P)$ and its initial probability distribution $\pi^{(0)}$.
Result: The limit probability distribution μ of the chain.

```
1  foreach transient state t do
2      μ_t ← 0;
3      foreach u ∈ S \ {t} do
4          π_u^(0) ← π_u^(0) + π_t^(0) P_{t,u} / (1 − P_{t,t})
5          foreach s ∈ S \ {t} do
6              P_{s,u} ← P_{s,u} + P_{s,t} P_{t,u} / (1 − P_{t,t})
7              P_{s,t} ← 0
8          end
9      end
10 end
11 foreach end component R_i do
12     Let π_{R_i}^(∞) = Σ_{r∈R_i} π_s^(0) and E_i be a system of linear equations;
13     foreach r ∈ R_i do  Add to E_i the equation μ_r = Σ_{r'∈R_i} μ_{r'} P_{r',r} ;
14     Solve the system E' under the condition Σ_{r∈R_i} μ_r = π_{R_i}^(∞) to obtain μ_r for each r ∈ R_i;
15 end
```

μ is 0 on each transient state: the behavior of the chain will eventually visit the end components and stay there forever.

If a chain has no transient states, the final probability $\pi_{R_i}^{(\infty)}$ of ending in end component R_i just corresponds to the sum of the initial probabilities of the states in R_i, since by definition of end component the behavior of the chain never exits from each end component. Note that for the same reason $\pi_{R_i}^{(\infty)} = \sum_{r \in R_i} \mu_r$.

If the chain has transient states, the loop in lines 1-10 of the algorithm trickles down the initial probability of the transient states appropriately to the end components. The transient states are removed one by one from the chain and the transition and initial probabilities of the other states updates accordingly. Let t be a transient state to be removed. For each other state $u \in S \setminus \{t\}$, the probability that the behavior of the chain will move to u after looping an arbitrary number of times on t is $\frac{P_{t,u}}{1-P_{t,t}}$. This means that whenever a path starts or passes by t, the next state it will visit after t will be u with probability $\frac{P_{t,u}}{1-P_{t,t}}$; for this reason, by redistributing the initial probability $\pi_t^{(0)}$ of t to each of its successor states u weighted by $\frac{P_{t,u}}{1-P_{t,t}}$ we do not change the long-term probability of visiting each end components of the chain. Note that this only holds because t is transient. This modification of the chain does decrease the convergence time, but we are not concerned by it in this algorithm.

Similarly, for each predecessor s and successor u of t we increase the probability of transitioning from s to u by $P_{s,t}\frac{P_{t,u}}{1-P_{t,t}}$ to account for the probability of the paths that pass by t. Finally we isolate t by setting $P_{s,t}$ to zero for each of its predecessors, effectively removing it from the chain.

The loop in lines 1-10 repeats this operation for each transient state in the chain, effectively removing them without changing the limit probabilities of visiting each end components. This parti of the algorithm is inspired by Algorithm 1 in [4]. Having redistributed the initial probability of the transient states to the end components, we can just compute the final probability $\pi_{R_i}^{(\infty)}$ of end component R_i as the sum of the initial probabilities of the states in R_i; this is done for each end component in line 12.

Since each R_i is irreducible we can compute the limiting distribution of each of them independ-

ently under the condition that $\sum_{r \in R_i} \mu_r = \pi_{R_i}^{(\infty)}$ by solving the system of equations

$$\forall r \in R_i.\ \mu_r = \sum_{r' \in R_i} \mu_{r'} P_{r',r}$$

Solving such system of equation provides the limit probability of each state in an end component, and doing it for each end component provides the limit probability of each state in the chain.

For the time complexity, the loop in lines 1-10 runs in time $O(|S|^3)$. For the system of equations in lines 11-15 we have to solve $O(|S|)$ systems of $O(|S|)$ equations. The Coppersmith-Winograd complexity for solving systems of n linear equations is $\sim O(n^{2.3})$, thus the final complexity is $\sim O(|S|^{3.3})$.

◀

Proof of Theorem 10. The algorithm uses a chain rule for mutual information. Let $\mathcal{X}$ and $\mathcal{Y}$ be two time-indexed Markovian stochastic processes. Then

$$I^{(t_1,t_2)}(\mathcal{X};\mathcal{Y}) = \sum_{i=t_1}^{t_2} H(X_i|X_{t_1},...,X_{i-1}) + H(Y_i|Y_{t_1},...,Y_{i-1}) -$$

$$-H(X_i,Y_i|X_{t_1},Y_{t_1},...,X_{i-1},Y_{i-1})$$

This can be proved as follows:

$$\begin{aligned}
I^{(t_1,t_2)}(\mathcal{X};\mathcal{Y}) &= H^{(t_1,t_2)}(X_i) + H^{(t_1,t_2)}(Y_i) - H^{(t_1,t_2)}(X_i,Y_i) = \\
&= \sum_{i=t_1}^{t_2} H(X_i|X_{t_1},...,X_{i-1}) + \sum_{i=t_1}^{t_2} H(Y_i|Y_{t_1},...,Y_{i-1}) - \\
&\quad - \left(\sum_{i=t_1}^{t_2} H(X_i|X_{t_1},...,X_{i-1},Y_{t_1},...,Y_{i-1}) + H(Y_i|X_{t_1},...,X_i,Y_{t_2},...,Y_{i-1}) \right) \\
&= \sum_{i=t_1}^{t_2} H(X_i|X_{t_2},...,X_{i-1}) + \sum_{i=t_1}^{t_2} H(Y_i|Y_{t_2},...,Y_{i-1}) - \\
&\quad - \sum_{i=t_1}^{t_2} H(X_i,Y_i|X_{t_1},Y_{t_1},...,X_{i-1},Y_{i-1}) \\
&= \sum_{i=t_1}^{t_2} H(X_i|X_{t_1},...,X_{i-1}) + H(Y_i|Y_{t_1},...,Y_{i-1}) - \\
&\quad - H(X_i,Y_i|X_{t_1},Y_{t_1},...,X_{i-1},Y_{i-1})
\end{aligned}$$

The algorithm starts by computing the mutual information at time t_1, then updates it step by step until time t_2. The correctness of an update step comes from considering that due to the Markov property it holds that

$$H(X_i|X_0,...,X_{i-1}) = H(X_i|X_{i-1}) = \sum_{s \in S} \pi_s^{(i-1)} L(s)$$

The algorithm clearly terminates for a finite t_2 and $|S|$. The computation of the marginals can be solved in $O(|S|^2)$, while the probability distributions at time t_1 require time $O(t_1|S|^2)$, dominating the cost of the other operations before the `for` cycle. Inside the cycle the operations have cost $O(|S|^2)$ and the cycle gets repeated $t_2 - t_1$ times, bringing the total cost to $O(t_2|S|^2)$ ◀

Proof of Theorem 11. We show a reduction to the bounded leakage problem from Skolem's Markov chain reachability problem.

Consider an instance of Skolem's Markov chain reachability problem, i.e. a Markov chain $\mathcal{C} = (S, s_0, P)$, a state s and a probability r. Label state s with the information $\mathtt{h} = 0, \mathtt{o} = 0$ and all other states with the information $\mathtt{h} = [0, 1], \mathtt{o} = 1$, for a secret variable $\mathtt{h}$ and an observable variable $\mathtt{o}$. Then being in state s gives us exactly 1 bit of information over $\mathtt{h}$, while no other state gives any information.

Let $I(\mathtt{o};\mathtt{h})^k$ be the mutual information between $\mathtt{o}$ and $\mathtt{h}$ at time $k \in \mathbb{N}$. Let p_k be a path in $\mathcal{C}$ of length k, $p_k(s)$ a path in $\mathcal{C}$ of length k ending in state s, and $p_k(\neg s)$ a path in $\mathcal{C}$ of length k not ending in state s. Then it holds that

$$
\begin{aligned}
I(\mathtt{o};\mathtt{h})^k &= H(\mathtt{h}) - H(\mathtt{h}|\mathtt{o}_1, ..., \mathtt{o}_k) && \text{(by def. of } I(\mathtt{o};\mathtt{h})^k\text{)} \\
&= 1 - H(\mathtt{h}|\mathtt{o}_1, ..., \mathtt{o}_k) && \text{(since } H(\mathtt{h}) = 1 \text{ by construction)} \\
&= 1 - \sum_{p_k \in S^k} P(p_k) H(\mathtt{h}|\mathtt{o}_1, ..., \mathtt{o}_k = p_k) \\
&= 1 - \sum_{p_k(\neg s) \in S} P(p_k(\neg s)) H(\mathtt{h}|\mathtt{o}_1, ..., \mathtt{o}_k = p_k(\neg s)) \\
& && \text{(since } H(\mathtt{h}|\mathtt{o}_1, ..., \mathtt{o}_k = p_k(s)) = 0\text{)} \\
&= 1 - \sum_{p_k(\neg s) \in S} P(p_k(\neg s)) && \text{(since } H(\mathtt{h}|\mathtt{o}_1, ..., \mathtt{o}_k = p_k(t)) = 1 \text{ for } t \neq s\text{)} \\
&= 1 - (1 - \sum_{p_k(s) \in S} P(p_k(s)) \\
& && \text{(since the events } X_k = s \text{ and } X_k \neq s \text{ are complementary)} \\
&= 1 - (1 - P^{(k)}_{s_0,s}) && \text{(by definition of } P^{(k)}_{s_0,s}\text{)} \\
&= P^{(k)}_{s_0,s}
\end{aligned}
$$

proving that the leakage at time k is equivalent to the reachability probability of state s at time k. Thus if we had a way to determine whether the leakage of the system at some time t is r bits we would conclude that $\pi^{(t)}_s = r$, deciding Skolem's Markov chain reachability problem. Thus we conclude that determining whether there is a time t such that the leakage of the system is a given value is at least as hard as deciding Skolem's problem. ◀