Dynamic Security Aspects of Onion Routing*

Alessandro Melloni [1], Martijn Stam [1], and Øyvind Ytrehus [1]

Simula UiB, Bergen, Norway.
{alessandro,martijn,oyvindy}@simula.no

Abstract. An *anonymous communication network (ACN)* is designed to protect the identities of two parties communicating through it, even if an adversary controls or observes parts of the network. Among the ACNs, Tor represents a practical trade-off between offering a reasonable level of anonymity and, simultaneously, an acceptable transmission delay. Due to its practical impact, there is abundant literature on the performance of Tor concerning both communication and security aspects.
Recently, a static framework was suggested for evaluating and comparing, in a quantifiable way, the effect of different scenarios (attacks, defence mechanisms, and other protocol changes). Although a static model is useful, many scenarios involve parameters and stochastic variables that change or evolve over time, or that may be influenced by active and malicious adversaries. In this paper, we propose a *dynamic* framework for evaluating such scenarios. We identify several scenarios where this framework is applicable, and illustrate our framework by considering the *guard node* mechanism in Tor. We evaluate and compare variations on the guard node concept suggested in the literature with respect to relevant performance metrics and, using the framework, support our evaluation with a theoretical analysis.

Keywords: Anonymity · Onion Routing · Tor · Traffic Analysis

1 Introduction

Onion routing aims to provide anonymity by obfuscating the link between a user and their network destination [17,18,39,43]. Ideally, the link remains hidden even against adversaries who observe or influence large swaths of the network. The most widespread implementation of onion routing is Tor [13], which relies on users picking multiple nodes from the network and establishing *circuits* to relay traffic through the nodes. These nodes are referred to as *onion routers*, and their identities are collected and distributed to the users by a central authority.

The anonymity provided by a Tor circuit strongly depends on what an adversary can observe. If all routers on the circuit are adversarially controlled, no anonymity is possible. Moreover, the first and last node on a circuit play a crucial role as the first node can easily identify the user whereas the last node knows

*This is the full version of the paper that appears in the proceedings of the 19th IMA International Conference on Cryptography and Coding.

the destination. If an adversary can correlate the two, e.g., by traffic analysis, anonymity is lost. Furthermore, an adversary only observing the first node can still attempt website fingerprinting to infer the destination.

Indeed, since its inception, these and other attacks, as well as improvements to Tor to counter them, have been proposed (see [25] for a recent overview). Whereas some attacks, such as traffic analysis and website fingerprinting, can be cast in a static framework [35], more advanced adversaries and countermeasures require a more dynamic framework to evaluate and compare threat models and protocol modifications. Let us elaborate with four illustrative scenarios.

Guard Nodes. In recognition of the importance of the entry node and its honesty in providing anonymity, *guard nodes* were introduced through a series of modifications to the original Tor design [30, 33]. Based on previous research [14, 20, 37, 45], the underlying philosophy is to improve the anonymity for the majority of users by sacrificing that of a few. Or, as Spock would say: "the needs of the many outweigh the needs of the few" [41].

Although guard nodes are initially randomly selected by a user, they become the choice of entry node for that user's circuits across an extended period of time. In this way, users who pick an honest entry node will be "guaranteed" to be safe for an extended time frame, compared to always choosing a new entry node for every circuit. In contrast, users picking a corrupted guard node will suffer from a security degradation as more of their circuits will be exposed. Hence, analysing the security trade-offs provided by guard nodes necessarily involves the modelling of circuit re-establishment, which is a dynamic feature.

Dynamic Unlinkability. Dynamic (un)linkability relates to an adversary trying to link together multiple subsequent sessions by the same user, without necessarily identifying that user [27, notion $(\mathsf{2S})\,\overline{\mathsf{L}}$]. It is a relevant goal for privacy enhancing technologies like Tor as it prevents the profiling of users over time, hindering widespread attempts of targeting specific users based on their past activity [5, 8, 15, 29]. There is a clear tension between this privacy goal and the guard node feature, as the use of a fixed entry node might be used to track users over time [10].

Circuit Lifetimes in Tor. A related issue is the influence on users' privacy due to the lifetime of circuits in Tor [26]. Currently, after 10 minutes from the first use, circuits are kept open only for already existing streams, and new streams are attached to new circuits. Similarly to the guard nodes feature, an analysis of the effects of different circuit lifetimes requires inherently a dynamic approach that allows for evolution of the Tor network over time.

Tagging Attacks. An adversary controlling both the entry and the exit node of a circuit, can use traffic analysis to deanonymise the circuit with high likelihood. Certainty can be achieved using tagging attacks [16, 38], where an adversary tampers with the cells at the entry node and then undoes the tampering at the exit node, deterministically. If the cells were indeed on the same circuit, the

malleability of the relay encryption in the current Tor specification [1, 9] will result in circuit behaviour as if no tampering took place (and deanonymisation has succeeded). However, if the adversary does not control the exit node, the tampering will cause the honest exit node to drop the circuit, likely triggering the user into the establishment of a new one [12, Section 5.4]. Although the confirmation aspect of successful tagging attacks could conceivably be studied using the static framework, the powerful consequence of triggering circuit re-establishment for not-yet-successful ones necessitates a dynamic perspective [44].

Our Contribution. We provide an evaluation framework to evaluate the efficacy of attacks by adversaries observing and possibly interacting with the Tor network over time. The framework can be applied to a variety of aspects of the onion routing protocol, including for instance:

1. effects of the guard nodes feature on anonymity;
2. guarantees of dynamic unlinkability;
3. severity of tagging attacks.

Our dynamic framework enables the comparison of different attacks, threat models, and metrics in the dynamic scenario, and facilitates the discovery and identification of gaps in the literature. Such gaps may occur, for example, in the cases where different attacks (or defences against them) are published and evaluated with the use of mutually incompatible metrics.

Related Work. Since its inception, several analyses of Tor's anonymity have been conducted [2–4, 27, 28, 34] (see also [32, Section 2] for a comprehensive account of different frameworks). A common approach, inspired by cryptographic games and proofs of security, is to formally define a game, the adversary, and a challenge. Main drawback of this approach is the lack of time-dependent and dynamic features being captured in the framework, as highlighted also by Backes et al. [4]. They combined the formal approach with a concept of "time" within the framework, but focused on timing as additional information available to the adversary (e.g., for traffic analysis) instead of the evolution of the network itself over time. On the other hand, simulation-based approaches [22, 23] allow for more practical analyses, with the distinct challenge of isolating and analysing the impact that the many parameters have on the results of the experiment.

2 Preliminaries

Notation. We use capitalised letters $A, B, \ldots$ to refer to random variables, and we base our treatment on the relationships among them. In particular, the causal relationship, denoted $A \rightsquigarrow B$, indicates that in a real-world system random variable A will be set before B and $\mathrm{I}(A; B) > 0$; in other words, the random process modelled by the variable A happens before the process modelled by B and the outcome of the former influences the outcome of the latter. We use $\mathcal{U}$

and $\mathcal{C}$ to indicate the sets of users and, respectively, circuits; the boldface letters $\mathbf{u}$ and $\mathbf{n}$ refer to the numbers of users and guard nodes.

Let X, Y be discrete stochastic variables with outcomes in $\mathcal{X}$ and $\mathcal{Y}$, respectively. We remind the reader that the *entropy* of X is [6]

$$\mathrm{H}(X) \triangleq \mathbb{E}[-\log X] = -\sum_{x\in\mathcal{X}} f_X(x)\log f_X(x), \tag{1}$$

and the *conditional entropy* of X given Y is

$$\mathrm{H}(X \mid Y) = -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} f_{XY}(x,y)\log f_{X|Y}(x|y). \tag{2}$$

where $f_X(x)$ is the probability mass function for X and $f_{XY}(x,y)$ is the joint probability mass function for (X,Y). The *mutual information* between X and Y is defined as

$$\mathrm{I}(X;Y) = \mathrm{H}(X) - \mathrm{H}(X \mid Y) = \mathrm{H}(Y) - \mathrm{H}(Y \mid X). \tag{3}$$

Random variables X and Y are independent iff $\mathrm{H}(X) = \mathrm{H}(X \mid Y)$, while if the value of X can be determined precisely after observing the value of Y, $\mathrm{I}(X;Y) = \mathrm{H}(X)$.

2.1 The Static Framework

Melloni, Stam, and Ytrehus [35] introduced a novel framework for assessing the security of low-latency ACNs in static scenarios such as traffic analysis and website fingerprinting. This *static framework* deviates from prevailing, more rigid approaches [21, 27] by incorporating security metrics beyond conventional adversarial advantages and drawing inspiration from cryptographic games.

Central to the framework are random variables that capture different aspects of the game. These variables describe general behaviour of ACNs, and their purpose is to enable the identification of potential information leakages and quantify these in terms of conditional entropy and mutual information.

At the core of the framework sits the random variable S, which models the users connecting to their chosen destinations; S represents the secret an adversary wants to uncover. The random variable G represents all information in the system that could possibly be observed. For instance, in Tor it contains the states of all proxies (users) and routers, as well as all sorts of traffic traces.

In general, the precise distribution of G is hard to pin down as it depends on a lot of factors, e.g. the length of circuits, how nodes are chosen, protocol specification, as well as general load on the ACN and its underlying infrastructure. However, G uniquely determines S in the sense that $\mathrm{H}(S \mid G) = 0$.

The view of the adversary, V, captures the information (from G) that is actually available to the adversary: Once the threat model is fixed, V is completely determined by G, so $\mathrm{H}(V \mid G) = 0$.

The goal of the adversary is specified by a query $\mathtt{q}$, that identifies partial information on S of particular interest. The adversary processes its view V into an output O that is relevant for the query $\mathtt{q}$ at hand. For instance, it could be the adversary's posterior best guess for the answer to $\mathtt{q}$. In any case, in the literature this processing is usually referred to as an *attack* on Tor and metrics are used to measure how well the attack actually fared.

Even though the exact probability distributions of the random variables may be unknown, their relationship is clear and can be defined in terms of information-theoretic notions, in particular using conditional entropy as done above. Moreover, as O arises from data processing of V, $\mathrm{H}(O \mid V) = 0$ holds and the data-processing inequality [6, Theorem 2.8.1] implies that $\mathrm{I}(V; S) \geq \mathrm{I}(O; S)$. Note that these conditions coexist with the causality relations $S \rightsquigarrow G \rightsquigarrow V \rightsquigarrow O$.

Our description of the static framework so far contained one small simplification. Melloni et al. additionally introduce a random variable Z representing auxiliary information gathered by an adversary during an initial training phase that can subsequently be used to refine the actual attack, that is the processing of V into O. For instance, V typically includes traffic traces and their shaping may be heavily affected by the destination or network topology, irrespective of the identity or even behaviour of the user. The variable Z captures what an adversary learns about the random behaviour of the ACN independent of the secret S. Thus $\mathrm{I}(S; Z) = 0$, yet the information might be useful for the processing, in the sense that $\mathrm{H}(S \mid V, Z) \leq \mathrm{H}(S \mid V)$ or, equivalently, $\mathrm{I}(S; V, Z) \geq \mathrm{I}(S; V)$.

Melloni et al. suggest Z might also contain an estimate of how S is distributed, for instance an adversary might try to determine the rough popularity of websites prior to any attack. To capture such a scenario formally, one could take a Bayesian approach where S is distributed according to a fixed model with unknown parameters (themselves following an uninformative prior); in that case Z can contain an estimation of the parameters according to which S is distributed. As we are primarily interested in evaluation scenarios, as opposed to real-life attacks, we can always assume the distribution of S is fixed and known.

Perspectives. The static framework is especially useful to specify an experimental setup used to evaluate security. In that case, a simplified scenario may be studied in lieu of a more complicated, realistic model and identifying the simplifications and their justifications (and limitations) can be useful. Often different metrics are required depending on the perspective that is taken, for instance a user might be primarily concerned about the likelihood of being deanonymised (a worst case), whereas a designer might be more interested in the expected number of compromises (an average case). Some simplifications will be suitable for one perspective, but perhaps less so for another.

3 A Dynamic Framework

Although the static framework can model attacks that involve time, such as traffic analysis, it is ill-equipped to deal with situations where parties actively

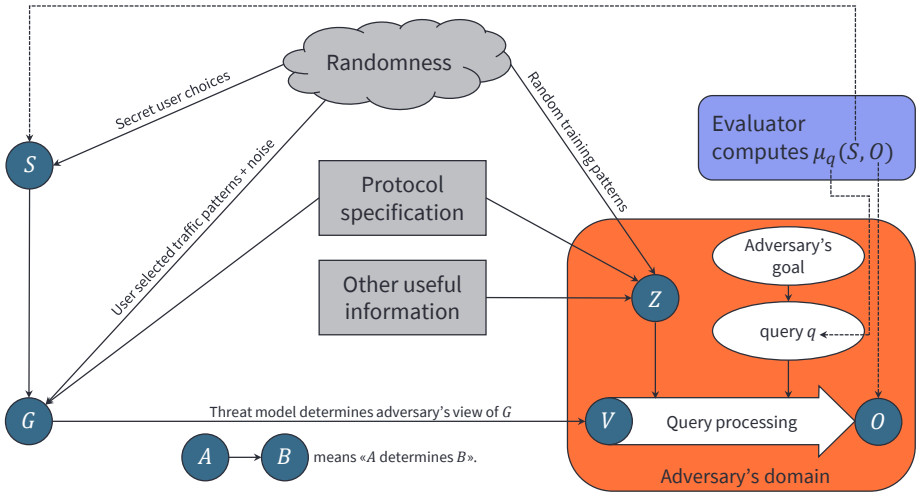


Fig. 1. Relations among the random variables in the static framework. Reproduced from Melloni et al. [35].

change their behaviour over time, or where the system design introduces dependencies over time. To address such scenarios, in this section we introduce a general framework for an analysis of ACNs involving dynamic behaviour. Similar to the static framework, the dynamic one is described in terms of general random variables and the relations between them, where we focus on those relations that best capture dynamic aspects of the game. When applying the framework to a concrete scenario, the random variables and their relations should be specified in sufficient detail to answer any relevant research question, for instance what spaces the various random variables are defined over and how they are distributed. Irrelevant details, not pertinent to the research question can be abstracted away or left un(der)specified.

Introducing Epochs. To enable research into the effects of active adversaries tampering with the network as well as dynamic features of the onion protocol, we allow evolution of the environment by introducing the concept of epochs. An epoch is a period of time during which we assume behaviour by the parties to be fixed, so within an epoch the static framework applies. A dynamic picture emerges by considering a sequence of epochs, effectively discretising real time. Indeed, for each epoch $\mathtt{t} \in \mathbb{N}$ we will consider the random variables $S_\mathtt{t}$, $G_\mathtt{t}$, $V_\mathtt{t}$, $O_\mathtt{t}$, and $Z_\mathtt{t}$. Additionally, we will for a given random variable $X_\mathtt{t}$, let $X^\mathtt{t}$ denote the collection of all the previous X_i up to and including $\mathtt{t}$.

In the dynamic setting, the focus of an adversary's goal with respect to the epoch $\mathtt{t}$ can be made explicit by writing $\mathtt{q_t}$ for the adversarial query and $\mu_\mathtt{q}(\mathtt{t})$ for a corresponding evaluation metric. The notation $\mu_\mathtt{q}(\mathtt{t})$ emphasizes the metric

as a function over time; in Sections 4 and 5 we will consider different metrics μ_i, subscripted simply by an index i, as they all relate to the same query $\mathtt{q}$.

Cross-epoch Variable Dependencies. Within a given epoch $\mathtt{t}$, the way the variables $S_\mathtt{t}$, $G_\mathtt{t}$, $V_\mathtt{t}$, $O_\mathtt{t}$, and $Z_\mathtt{t}$ relate to each other is directly inherited from the static framework. For the dynamic framework, we are interested in plausible relationships between variables from different epochs. Here we make a further distinction between intra-variable dependencies (e.g. between $S_\mathtt{t}$ and $S^{\mathtt{t}-1}$) and inter-variable dependencies (e.g. between $S_\mathtt{t}$ and $G^{\mathtt{t}-1}$).

We identify several possible scenarios, mainly depending on whether and how the dependency manifests for the different random variables. Here we concentrate on the core of the dependency, for instance when considering how $G_\mathtt{t}$ depends on $G^{\mathtt{t}-1}$, we are less interested in the logical consequences in any dependency that can be (fully) explained by $S_\mathtt{t}$ depending on $S^{\mathtt{t}-1}$.

Intra-variable Dependencies. Consider how users select their destinations, as captured by $S_\mathtt{t}$. As the boundary between two epochs is somewhat arbitrary when considering real-life Tor, some users are likely to stick to a destination, whereas others might change (or connect for the first time). Thus, a user's future behaviour might depend on the present, but arguably not on the past, so $S_\mathtt{t}$ satisfies the (first order) Markov property, namely that $\mathrm{I}\big(S_{\mathtt{t}+1};\, S^{\mathtt{t}}\big) = \mathrm{I}\big(S_{\mathtt{t}+1};\, S_\mathtt{t}\big)$.

For evaluation purposes, it can be useful to restrict to one of the two extreme cases: either the outcome is drawn once and then fixed, so $S_{\mathtt{t}+1} = S_\mathtt{t}$ which implies that $\mathrm{I}\big(S_{\mathtt{t}+1};\, S_\mathtt{t}\big)$ is maximal; or the different epochs are independent of each other, so $\mathrm{I}\big(S_{\mathtt{t}+1};\, S_\mathtt{t}\big) = 0$ and minimal. The former, fixed case is useful to analyse active attacks exploiting adversarially triggered circuit tear-downs (including tagging attacks), whereas the latter case can be used to analyse linkability over longer time domains. In a real-life setting, the mutual information lies somewhere between these extremes.

When considering $G_\mathtt{t}$, we recall that according to Tor protocol specifications, each user and router participating in the network is stateful and maintains all relevant information about its circuits and connections in the current state [12]. Thus, for circuits that remain active from one epoch to the next, $G_{\mathtt{t}+1}$ will depend on $G_\mathtt{t}$, without any further dependency on past epochs. Similarly, for guard nodes, any state maintained by a user regarding the identity (and usage) of guard nodes will be part of $G_\mathtt{t}$. Consequently, without significant loss of applicability, we can assume that also $G_\mathtt{t}$ exhibits a first order Markov property, i.e. $\mathrm{I}\big(G_{\mathtt{t}+1};\, G^{\mathtt{t}}\big) = \mathrm{I}\big(G_{\mathtt{t}+1};\, G_\mathtt{t}\big)$.

As the adversary's view $V_\mathtt{t}$ is a function of $G_\mathtt{t}$, it might inherit its Markov property. In the static model, the threat model ruling how V is a function of G can be deemed fixed; in the dynamic model, an adversary might corrupt different routers in different epochs, thus changing which part of $G_\mathtt{t}$ is visible in $V_\mathtt{t}$.

For an adversary's output $O_\mathtt{t}$ the question of how it relates to its predecessors $O_{\mathtt{t}-1}$ is, to a large extent, moot. What matters is that, on the one hand, $O_\mathtt{t}$ may well depend on the adversary's view on all epochs so far, i.e. on $V^{\mathtt{t}}$ and not just

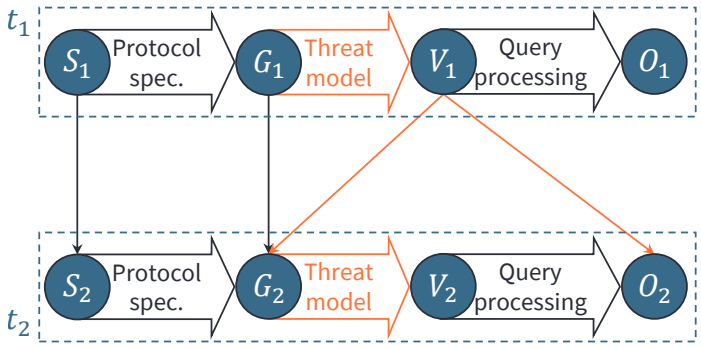


Fig. 2. Relations among the random variables from epoch $\mathtt{t}_1$ to epoch $\mathtt{t}_2$.

$V_\mathtt{t}$. For instance, when trying to link users' information over multiple epochs, an adversary will necessarily have to combine its view across epochs. On the other hand, we might be interested in how an adversary's success evolves as a function of time $\mathtt{t}$. However, in that case the appropriate tool is a metric $\mu_\mathtt{q}(\mathtt{t})$ rather than a direct statement on $O_\mathtt{t}$ itself.

Finally, for the auxiliary random variable $Z_\mathtt{t}$, we assume the adversary to accumulate all the collected information, so $\mathrm{H}\big(Z_\mathtt{t} \mid Z_{\mathtt{t}+1}\big) = 0$ for $\mathtt{t} \in \mathbb{N}$.

Inter-variable Dependencies. We already identified one inter-variable dependency above, namely in situations where an adversary's output $O_\mathtt{t}$ may depend on the view $V^\mathtt{t}$ across all epochs so far in case the goal is epoch-spanning. Yet, even if the goal is specific to the current epoch, if $S_\mathtt{t}$ is partially dependent on $S_{\mathtt{t}-1}$, then information collected in the previous epoch might still serve an adversary well, for instance to rule out some destinations from the pool of possible ones (improving the confidence level in the output $O_\mathtt{t}$).

Another inter-variable dependency arises when an adversary uses the view in $V_\mathtt{t}$ to disrupt the network, which will subsequently be reflected in $G_{\mathtt{t}+1}$. For instance, an adversary might block access to some routers, thereby influencing how users pick the routers for their circuits.

Summary. We considered possible relations among the different random variables across epochs, often expressed using information-theoretic notions. In Fig. 2 we provide an overview of these relations, with the arrows highlighting *possible* influences. Here the influence of the adversary is denoted using the orange colour: for example, based on the view V_1 and the threat model [35, Section 5], the adversary may choose to disrupt some circuits in G_2.

The absence of direct arrows between two random variables does not mean that they are independent, i.e. their mutual information can be non-zero: rather, it highlights that their dependency is already captured by the relation between other random variables at a higher level in the chain. For example, there is no

arrow from S_1 to G_2 because we assume that all information from S_1 influencing G_2 is also contained in S_2, i.e. $\mathrm{I}(G_2;\, S_2) = \mathrm{I}(G_2;\, S_1, S_2)$.

4 Application of the Framework to Guard Nodes

In this section we discuss how to apply the dynamic framework to the study of the performance of guard node scenarios.

Context. Guard nodes were introduced in 2013 [33], recognizing that *"some circuits are going to be compromised, but it's better to increase your probability of having no compromised circuits at the expense of also increasing the proportion of your circuits that will be compromised if any of them are."* The underlying technical assumption is that a single party controlling both the first and last nodes of a circuit can link source and destination of the traffic, and a single compromised circuit may suffice to ruin a user's anonymity. A further, more philosophical assumption is that, once one circuit is compromised, the additional damage of further compromised circuits to a user's anonymity is limited.

Fixing the first node over a period of time increases the probability of having no compromised circuits over that period of time when compared to randomly picking a new entry node every new circuit setup. This fixed entry node is known as a *guard node*. In practice, users will still need to change guard nodes from time to time, for instance when their guard node is overloaded or unavailable, the last referred to as the natural *churn* of the network. To balance these unpredictable events, the guard node feature employed by Tor shortlists several nodes and then selects the guard node from this list; after a while, the list is refreshed [31].

As a result of this mechanism, the guard node policy actually consists of two distinct selection parts. First, a *guard list maintenance policy* describes how to construct the list of potential guard nodes, how many to pick, and when to pick new ones: this refresh process is referred to as *guard rotation*. Second, a *guard selection policy* dictates which node to pick from the short list whenever the proxy builds a new circuit. It is noteworthy to remark that literature on the subject tends to disregard the distinction between guard list maintenance policy and guard selection policy, focusing almost exclusively on the first.

The guard list maintenance policy is influenced by two factors: the churn of the nodes happening in the network and the guard rotation defined in the policy itself. Guard rotation also allows for recovery after compromise, as unlucky users picking corrupted guard nodes will refresh them after their lifetime expires.

Several changes [10, 14, 19, 20, 30, 42] to the original guard list maintenance algorithm have been proposed and analysed. These proposals mainly investigate the effects of changing the lifetime and quantity of guard nodes, both on performance and security. The fact that fixing a single guard node maximises security has already been recognised [10], but with some caveats when considering that nodes might be unreachable [14].

Security Evaluation. When the guard node feature was announced [33], the goal stated by the Tor team was to decrease the *number of deanonymised users*, conceding a higher *number of uncovered destinations for the users that have been deanonymised*. The first two security metrics we will consider are focused at estimating these quantities.

Later, Johnson et al. [24] employed the probability distribution of the *time until first compromise* as a security metric. In this paper, for ease of presentation we will instead use the average time until first compromise as our third metric. Lastly, we will consider a fourth metric to describe the situation for a user who is among the unlucky ones. We will describe these metrics in more detail in the **Metrics** subsection, after we have discussed the dynamic framework in the context of this specific application.

Security evaluation depends also greatly on the specific definition of *compromise*: different authors employ different approaches, possibly affecting a fair comparison. For example, Hayes and Danezis [20] consider a user to be compromised the moment they choose a malicious guard node (even before using it), while for Johnson et al. [24] a user needs to actually build a circuit through a malicious node to be compromised. We discuss this further in Section 5.1.

Our evaluation framework allows us to reveal hidden assumptions and simplifications (Section 5.1), facilitating both an exploration of possible alternatives and a combinatorial analysis and comparison of various guard node policies across scenarios (Section 5.2). It also enables a unified comparison of approaches discussed in the literature. We illustrate this with Example 1.

Remark 1. When considering the security offered by guard nodes, the emphasis is usually on circuit compromise. In our analysis, we will follow this lead, however there are other security ramifications tied to the use of guard nodes. For instance, it might be possible to identify a user based on their chosen guards (guard node fingerprinting [10]), which could violate the privacy goal of unlinkability of sessions over an extended period of time. The above holds even if each circuit is selected with an a honest exit node. In addition to guard node fingerprinting, ingress traffic fingerprinting allows linking user across epochs.

Dynamic Modelling. To model guard nodes in our framework, we start by specifying a minimal setup of the random variables that suffices to capture the various guard node policies and their intended effect on security.

System Setup. We will consider $\mathbf{u}$ users that each connect to a single destination $d_{u,\mathtt{t}}$ per epoch $\mathtt{t}$, where we furthermore assume that users select their destinations independently of each other and different users may select their destination with different probability distributions. These choices specify the secrets $S_{\mathtt{t}}$, where additionally we assume $S_{\mathtt{t}}$ to be independent across epochs. From the users' perspective, each epoch is marked by the setup of their respective circuits.

The variable $G_{\mathtt{t}}$ contains the state of the system and possible observables for an adversary. We assume that during each epoch, every user establishes a new circuit to its destination, resulting in the set of circuits $\mathcal{C}$. Sticking to

the default Tor circuit length of 3, each circuit can be represented as $c_i = (c_{\mathtt{ID}}, u_{\mathtt{ID}}, g_{\mathtt{ID}}, m_{\mathtt{ID}}, e_{\mathtt{ID}}, d_{\mathtt{ID}})$, consisting of $\mathtt{ID}$ for the circuit itself, the user, guard, middle, exit nodes and destination. Here the circuit identity $c_{\mathtt{ID}}$ is simply a global identifier used *in the framework.*

Given the circuit belonging to user u in epoch $\mathtt{t}$, the destination $d_{\mathtt{ID}}$ will match that in $S_{\mathtt{t}}$, so equal $d_{u,\mathtt{t}}$. The middle router $m_{\mathtt{ID}}$ and exit router $e_{\mathtt{ID}}$ are assumed to be chosen independently and uniformly at random, whereas, crucially, the guard node $g_{\mathtt{ID}}$ is selected according to the guard node policy.

Remark 2. The "uniform i.i.d." assumption on $m_{\mathtt{ID}}$ and $e_{\mathtt{ID}}$ is a simplification of the real-life scenario where bandwidth considerations—and for exit nodes possibly port support—come into play (deviating from uniformity); moreover, certain combinations of routers, for instance those all belonging to the same autonomous system, are avoided (introducing some dependency) [11, Section 2.2].

The guard node policy itself is modelled by maintaining (in $G_{\mathtt{t}}$) on the one hand all information that proxies require to select their guard nodes (such as bandwidth and availability) and on the other, for each proxy, that proxy's state pertaining to its guard nodes, as prescribed by the specific policy (for instance, the priority of the guard nodes, how long they have been in use by that proxy, etc.). Different guard selection proposals may require additional information, such as the mapping of guard nodes to their set in the *guard sets* proposal [20].

The guard node feature advantage starts from the second circuit setup and involves information from the previous epoch that is maintained into the current one, consistent with $G_{\mathtt{t}}$ being a Markov process. In case there is no guard node feature, $G_{\mathtt{t}}$ is memoryless and users select new entry nodes every circuit setup.

For the possible observables and for each node, operational information is maintained in $G_{\mathtt{t}}$ for all the circuits routed by that node. Here we use the model common for traffic analysis and website fingerprinting, and let guard and exit nodes observe $(u_{\mathtt{ID}}, m_{\mathtt{ID}}, \text{ingress trace})$ and $(m_{\mathtt{ID}}, d_{\mathtt{ID}}, \text{egress trace})$, respectively.

The Adversary. The view of the adversary $V_{\mathtt{t}}$ consists of a selection of the last two types of tuples. Which tuples an adversary can observe depends on the threat model, more specifically on which nodes are corrupted; we indicate the ratio of malicious entry and exit nodes with γ and ε respectively. Here one could make a distinction, as Melloni et al. [35] do, between full control where the adversary has completely corrupted the node and can see its state (including the identity of the middle router $m_{\mathtt{ID}}$ for every circuit through that node); and partial control where an adversary can observe the link traffic between the node and the outside world (so $u_{\mathtt{ID}}$ and ingress trace, respectively $d_{\mathtt{ID}}$ and egress trace, but not $m_{\mathtt{ID}}$). The difference between these two types of adversarial views relates to the amount of information acquired by the adversary *per time unit*, so that an adversary with the weaker level of node compromise may have to observe more traffic to obtain the same results. Pragmatically, the difference between these two cases can be modelled by different ε values (where ε is changed to mean "the probability of full circuit deanonymisation given a guard compromise".)

In these minimal settings, the adversary aims to uncover as many user–destination pairs as possible, based on V_{t}.|[†]
Since each user only has one destination per epoch, the adversary can simply output $O_{\mathsf{t}} = \left\{\widehat{d}_{u_j,\mathsf{t}}\right\}$, indicating the guessed destination for users $u_1, \ldots, u_i$ in epoch t.

Metrics. To formalize the security metrics mentioned earlier, we introduce two auxiliary random variables: $X_{\leq \mathsf{t}}(u)$ represents the number of correct guesses by the adversary up to epoch t for user u, defined as $X_{\leq \mathsf{t}}(u) = \sum_{i=1}^{\mathsf{t}} \left[\widehat{d}_{u,i} \stackrel{?}{=} d_{u,i}\right]$; and $Y_{\leq \mathsf{t}}$ denotes the number of users that have never been deanonymised (up to epoch t), i.e. $Y_{\leq \mathsf{t}} = |\{u | X_{\leq \mathsf{t}}(u) = 0\}|$.

The goal of the guard nodes feature is to minimize the deanonymised users:

$$\mu_1(\mathsf{t}) = \mathbb{E}\left[Y_{\leq \mathsf{t}}\right], \text{ or equivalently with } \mu_1(\mathsf{t}) = \sum_u \Pr[X_{\leq \mathsf{t}}(u) = 0]. \tag{4}$$

The "price" is an increase in deanonymised destinations for compromised users:

$$\mu_2(\mathsf{t}) = \frac{1}{\mathbf{u}} \sum_u \mathbb{E}[X_{\leq \mathsf{t}}(u) \mid X_{\leq \mathsf{t}}(u) > 0]. \tag{5}$$

We also simplify the metrics of by Johnson et al. [24], considering the average instead of the full distribution:

$$\mu_3 = \frac{1}{\mathbf{u}} \sum_u \mathbb{E}[\min\{\mathsf{t} | X_{\leq \mathsf{t}}(u) > 0\}]. \tag{6}$$

Recalling that the guard node construction gains anonymity for the majority by sacrificing a few, it is also useful to have a metric that quantifies how much the few will suffer. We will use the following metric:

$$\mu_4(\mathsf{t}) = \frac{1}{\mathbf{u}} \sum_u \Pr[X_{\leq \mathsf{t}+1}(u) > X_{\leq \mathsf{t}}(u) \mid X_{\leq \mathsf{t}}(u) > X_{\leq \mathsf{t}-1}(u)\,]. \tag{7}$$

4.1 Guard Nodes Policies

The framework allows us to compare several variations of guard node policies through computation of the metrics, but we first need to identify the parameters needed to describe a guard node policy. The number of nodes picked by each proxy, according to a defined probability distribution Δ, is $\mathbf{n}$. We also need an ordering $\preceq$ to specify the relative preference of guard nodes to use each time the user wants to setup a new circuit, and the maximum lifetime of T epochs for guard nodes before being refreshed.

[†]Alternatively, the goal can be to create, for each user, a list of possible destinations with associated probabilities, (thus allowing an approximation of the mutual information $\mathrm{I}(S; V, Z)$.

Table 1. Description of the identified metrics.

Metric	Description
$\mu_1(\mathsf{t})$	Expected number of users that, up to epoch t, have never been deanonymised (4).
$\mu_2(\mathsf{t})$	Expected number of uncovered destinations for any user that has been already deanonymised (5).
μ_3	Average time until first deanonymisation (6).
$\mu_4(\mathsf{t})$	Average of Pr[compromise at time $\mathsf{t}+1$ | compromise at time t] (7).

No Guard Nodes. The base case is having no guard nodes feature at all: the proxy chooses a new entry node for every circuit setup, i.e. $\mathsf{T} = 1$; we refer to this as no-guards policy.

Single Guard. The simplest guard based policy is the one with a single guard node, using only that for circuit setups until it expires or if it is unreachable. Formally, $\mathbf{n} = 1$ and $\mathsf{T} > 1$.

3-guards. A more practical policy is represented by the 3-guards policy, where the proxy selects 3 entry nodes and uses them as guard nodes until they expire or become unreachable. This guard policy is a simplified version of the current Tor Guard Specification (see below). The introduction of multiple guard nodes prompts the need to specify the guard selection policy, ruling which guard node to select for each circuit creation.

Tor Guard Specification. The current Tor guard node specification [31] is based on a series of subsequent samplings performed by the proxies, starting from the set of all the current guard nodes according to some probability distribution and some further processing (see Appendix A). The proxy picks, at run-time, 3 nodes to use as guards from a persistent short list, with nodes being removed from it when they are either unreachable for some epochs or their lifetime as guards expired.

Example 1 (COGS). Elahi et al. [14] analysed several parameters influencing the guard selection algorithm, such as adversarial bandwidth, natural churn of entry nodes, guard rotation and number of chosen nodes. They introduce a malicious node with fixed bandwidth, and estimate the ratio of circuits choosing it as guard node using historical data on the Tor network while varying the aforementioned parameters.

Modelling these setting into our framework requires specifying the random variables and the threat model. The outcome of the secret S_{t} is fixed and it consists of $\mathbf{u} = 80,000$ users, all of whom create a single circuit per epoch,

Table 2. Different models of guard node policies.

Type	Policy	Description
Guard maintenance	No-guards	No guard nodes.
	1-guard	Single guard node.
	3-guards	3 guard nodes, simplified Tor current specification.
	Tor Guard Specification	The current Tor guard node specification [31].
Guard selection	Ideal uniform-use	See "Simplified policies" in Section 5.1.
	Greedy	See "Simplified policies" in Section 5.1.

from $\mathtt{t} = 1$ until the end of the experiment. The threat model is represented by a single node (entry-only) having fixed bandwidth, and is introduced in the second epoch only, to simulate the real-world scenario of already-running Tor network being infiltrated by malicious actors. As a consequence, all circuits in the first epoch are safe and, from $\mathtt{t} = 2$ onwards, users will choose the entry node based on the guard node policy applied for that scenario, such as sampling $\mathbf{n} = 10$ guard nodes instead of the standard $\mathbf{n} = 3$, allowing to compare the effect of the examined parameter. The adversarial output $O_{\mathtt{t}}$ lists the circuits passing through the malicious guard node, and the metric is simply the number of observed circuits by the adversary, i.e. $\left|O_{\mathtt{t}}\right|$. Note that the metric does not depend on the variable $S_{\mathtt{t}}$: this is due to a further simplification, as discussed in Section 5.1.

5 Analysis

In Section 5.1, we first explain the simplifications and assumptions we apply in order to design an analytical model *for evaluation purposes*. Subsequently, in Section 5.2, we derive quantitative values for the metrics in Table 1 based on the parameters already introduced (see also Table 3. As an alternative to the analytical results, we also created a simple simulation program that is described in Section 5.3. Finally, in Section 5.4 discusses suitable parameter ranges, and in Section 5.5 we provide numerical results.

5.1 Simplifications & Assumptions

Research on ACNs is seldom performed on real-world data and systems, for both practical and ethical reasons, e.g. the lack of available information, the intractable complexity of the live Tor network, the need to create reproducible results from controllable network states and parameter values, or the privacy intrusion of observing actual Tor communication. To overcome these limitations,

Table 3. Parameters for evaluation of anonymity. The parameters $\mathbf{n}$, T, and φ can be thought of as part of Z_t, and influence S_t and its random expression into G_t. The parameters γ and ε are convenient to express the adversaries' capabilities and their success rates in terms of the metrics $\mu_1, \ldots, \mu_4$.

Sym.	Description
$\mathbf{n}$	Number of guard nodes in the guard list.
T	Maximum lifetime of guard list in epochs.
φ	Probability that a guard node is unreachable at circuit setup time, due to churn in the network.
γ	Probability of guard compromise.
ε	Probability of selecting an exit node compromised by an adversary that compromises the guard node.

researchers introduce simplifications and assumptions when analysing some aspects of Tor. Here we summarize the simplifications we subsequently apply in Section 5.2.

S_t: Simple and uniform user behaviour. In each epoch, each user selects a destination independently of the others and connects to it. Although S^t, in our framework and in general, may be described by a bipartite graph connecting, on one side, a large set of (potential) users and an even larger set of (potential) servers or websites, from an evaluation point of view it makes sense to restrict this to a standardised and simpler graph. For instance, to consider all users to act simultaneously in each epoch connecting to a single destination is a modelling simplification, that (almost certainly) does not hold true in the real world, but which may still shed light on specific aspects of the behaviour of Tor. Also, there is a diversity of users, but in the perspective of a specific user, the most important performance criterion is the anonymity of that user, as a function of the user's behavioural parameters. It seems like a realistic assumption that users act independently, in which case the number of users is not very important in the context of our four security metrics. Concerning the restriction of one circuit per epoch: in practice most users will have zero circuits in most epochs, if the epochs are counted as fixed duration consecutive time slots.

Uniform node corruption and the relevant range of parameter values. Each guard node is corrupted with a fixed probability γ, and each exit node is corrupted with a fixed probability ε. The Tor nodes available for selection as entry and exit nodes may operate in different environments. Moreover, it is plausible that node parameters that influence the probability of selection (e.g., available bandwidth) may be correlated with the probability of compromise. However, this information is typically not available to the user. Hence it makes

sense to model the selection of entry (guard) and exit nodes as independent random processes.

However, the interpretation of γ should be "the proportion of all potential guard nodes which are compromised", whereas the interpretation of ε should be "the proportion of all exit guard nodes which are compromised *by the same adversary that compromises the entry node*". Hence it makes sense to consider practical only the cases where $1 \gg \gamma \gg \varepsilon > 0$.

Note that the expected amount of compromised circuits across the entire Tor network does not change, i.e. $\mathbb{E}[X_{\leq \mathtt{t}}(u)] = \gamma \cdot \varepsilon \cdot \mathtt{t}$, even when considering the guard node feature.

Uniform guard node churn. Each guard node experiences churn with a fixed probability φ.

What constitutes a "breach of anonymity"? The adversary will try to exploit all the data contained in $V_{\mathtt{t}}$ to breach a user's anonymity. The guard node, in particular, learns the IP address of the user('s proxy), but a compromised guard node can also collect traffic traces for each circuit. These traffic traces can be used for e.g., website fingerprinting. However, the computational work of distinguishing a website only by observing traffic that passes through an entry/guard node can be challenging, especially when the destination is not a common one [36]. On the other hand, an adversary that controls both the entry and the exit node can (1) easily filter traffic that passes through a shared middle node, and align traffic traces to identify a circuit, and then (2) establish a user-destination pair. In the rest of the paper we will apply two simplified definitions for "breach of anonymity", described in the following.

"Breach of anonymity" ≡ guard and exit node compromise. In Section 5.2 we develop numerical results for a model where "breach of anonymity" is defined as a simultaneous compromise of guard node and exit node, as done in previous works [24]. Note, however, that since node compromise is modelled as a stochastic variable, the (probability of) event of exit node compromise may also include all events that allows deanonymisation of the circuit, including exit node compromise but also the event of a traffic trace that reveals the destination.

A more conservative view: "Breach of anonymity" ≡ guard node compromise. In other works [10, 14], a different assumption is employed: users are considered deanonymised as soon as they choose a malicious guard node, instead of requiring the circuit to be compromised. This view may be relevant when the mere action of connection through Tor can create a problem for the user, or if the traffic trace can be assumed to reveal the destination with very high confidence. Note that all of the results in Section 5.2 can be adapted to this more conservative view by setting $\varepsilon = 1$.

Remark 3. These two stronger assumptions remove the dependency on $S_{\mathtt{t}}$ from the evaluation metrics, as they equate circuit, resp. guard node, compromise to

destination uncovering. Even though the rationale for the guard node feature is worded in terms of probability of circuit compromise, thanks to the assumption that circuit compromise is equivalent to deanonymisation, we can use the latter to estimate the former. □

Abstraction by ignoring parameters irrelevant to the computation of metrics values. The first step in the analysis is identifying the parameters of the models, balancing the needs for relevance and simplicity; we aim to determine and explicitly split parameters for the model from the parameters of the guard node policies themselves. In the first category, we identify:

- the total number of guard nodes $\mathbf{n}$;
- for each epoch, the *churn*, or the probability φ of a node to be offline;
- the probability γ of a guard node to be compromised, and the probability ε of an exit node to be compromised.

The first parameter can be set to a specific value, or obtained from the current Tor consensus. Main approaches to compute the other parameters include getting estimates from some of the recent Tor consensuses or other recent literature.

Simplified policies.

Guard list selection. Each time a guards list is renewed, guards are selected uniformly at random. The no-guards and the 1-guard scenarios are easy to analyse: the 1-guard case still requires a policy for selecting guard nodes, but the quality of the guard node selection can still be summarized in the single probability γ. In contrast, as described in Section 4.1, multi-guard-node scenarios also require a policy for selecting a guard node (from a guard node set) to use at each new circuit setup, and another policy for dealing with the unavailability situation: when *none* of the guard nodes are reachable at the time of circuit setup.

Selection of guard for each circuit. The Tor guard specification (Appendix A) is designed for achieving multiple goals. Prevention of deanonymisation is one of the goals, but a guard for a new circuit is selected from the guard set based on priorities which in turn also uses other criteria, including throughput (nodes are weighted by their bandwidth). In order to focus on the security metrics as discussed here, we abstract from Tor's priority-based policy and instead focus on two simplified models, the best-case *greedy* policy and the opposite and worst-case *ideal uniform use* policy. These will serve as upper and lower bounds for the Tor performance, based on Lemma 5 and Lemma 6. In the greedy policy, the user will always try to use guard nodes that have been *most* used in the past, to maximize (Equation (10)). In the ideal uniform use policy, the user will always try to use guard nodes that have been *least* used in the past, approaching a round-robin type of behaviour when there is no churn. In our context, these theoretical policies serve to obtain bounds on Tor's anonymisation performance.

Unavailability and reset policies. In an **n**-guard policy, if all **n** guard nodes are simultaneously unavailable in an epoch, service is interrupted unless the policy includes a strategy for resolving the problem immediately. Assuming the simple strategy of immediately resetting the entire guard set when this event occurs, we note that this may reduce the effective lifetime of guard sets to a value less than the parameter T. For this reason, T should not be set to a value higher than the expected number of epochs between events of loss of all guard nodes.

Simplification: Guard lists disjoint over time. This may not be true, considering Tor rules for selecting guard lists, although the number of potential guard nodes runs in the thousands. However, this assumption simplifies the discussion. Also, the guard node state of being compromised can change with time. If not, and if a guard node is indeed reused over several periods, the results in Section 5.2 will overestimate the success rate of the attacker.

5.2 Quantitative Formulas for Metrics

In this section, we provide formulas for the metrics introduced in Section 4 (see Table 1) for the guard node policies identified in Table 2 (except the Tor Guard policy, the performance of which is bounded by that of other policies).

We assume a secret S_{t} consisting of $\mathbf{u}$ users, each of them connecting to a single destination in each epoch (possibly with different probability distributions), and such that $\mathrm{I}(S_{\mathsf{t}+1}; S_{\mathsf{t}}) = 0$. Since all users behave in the same way with respect to circuit creation, we simplify the random variable $X_{\leq \mathsf{t}}(u)$ to the shorter form $X_{\leq \mathsf{t}}$.

Three general and basic lemmas. In Lemmas 1 to 3 we give general expressions for metrics $\mu_1, \ldots, \mu_3$ that are valid for all guard node policies and the user behaviour that we will discuss. These expressions still rely on further calculations, specific to each policy, that will be derived further later on for some policies.

Lemma 1 (Formulas for $\mu_1(\mathsf{t})$ for any guard node policy). *Assume that in each epoch, each user, independent from the others, selects a destination and establishes a new Tor circuit to it. Then*

$$\mu_1(\mathsf{t}) = \mathbb{E}\big[Y_{\leq \mathsf{t}}\big] = \mathbf{u} \cdot \Pr[X_{\leq \mathsf{t}}(u) = 0]. \tag{8}$$

Proof. Since each of the $\mathbf{u}$ users is independent of the others, we obtain a binomial distribution for $Y_{\leq \mathsf{t}}$, where not being deanonymised corresponds to the success case:

$$\Pr\big[Y_{\leq \mathsf{t}} = k\big] = \binom{\mathbf{u}}{k} \left(1 - \Pr[X_{\leq \mathsf{t}} = 0]\right)^{\mathbf{u}-k} \cdot \left(\Pr[X_{\leq \mathsf{t}} = 0]\right)^{k}.$$

The expectation of this binomial distribution is given by Equation (8). □

Lemma 2 (Formulas for $\mu_2(\mathsf{t})$ for any guard node policy). *Let S_{t} be as described in Section 4, and assume that for each circuit setup, the exit node is selected independently of the guard node. Then, for any guard node policy,*

$$\mu_2(\mathsf{t}) = \frac{\mathsf{t}\gamma\varepsilon}{1 - \Pr[X_{\leq \mathsf{t}} = 0]}. \tag{9}$$

Proof. Consider a set of j circuits, all using the same guard node. The expected number of compromised circuits is

$$j\big((1-\gamma)\cdot 0 + \gamma\cdot\varepsilon\big) = j\gamma\varepsilon.$$

Thus, the number of times a guard node is used is irrelevant to the average probability of circuit compromise, and Equation (9) follows. □

Lemma 3 (Formulas for μ_3 for any guard node policy). *Let S_{t} be as described in Section 4. Then, for any guard node policy,*

$$\mu_3 = \sum_{\tilde{\mathsf{t}}=1}^{\mathsf{t}} \left(\tilde{\mathsf{t}} \cdot \Pr\big[X_{\leq\tilde{\mathsf{t}}} = 1 \;\big|\; X_{\leq\tilde{\mathsf{t}}-1} = 0\big] \cdot \Pr\big[X_{\leq\tilde{\mathsf{t}}-1} = 0\big]\right). \tag{10}$$

Proof. Let C be the set epochs where a given user is deanonymised, so that $C = \{\mathsf{t} | X_{\leq\mathsf{t}} > 0\}$. Then

$$\begin{aligned}
\mathbb{E}[\min C] &= \sum_{\tilde{\mathsf{t}}=1}^{\mathsf{t}} \left(\tilde{\mathsf{t}} \cdot \Pr\big[\min C = \tilde{\mathsf{t}}\big]\right) \\
&= \sum_{\tilde{\mathsf{t}}=1}^{\mathsf{t}} \left(\tilde{\mathsf{t}} \cdot \Pr\big[X_{\leq\tilde{\mathsf{t}}-1} = 0 \wedge X_{\leq\tilde{\mathsf{t}}} = 1\big]\right) \\
&= \sum_{\tilde{\mathsf{t}}=1}^{\mathsf{t}} \left(\tilde{\mathsf{t}} \cdot \Pr\big[X_{\leq\tilde{\mathsf{t}}} = 1 \;\big|\; X_{\leq\tilde{\mathsf{t}}-1} = 0\big] \cdot \Pr\big[X_{\leq\tilde{\mathsf{t}}-1} = 0\big]\right).
\end{aligned}$$

□

The no-guards and 1-guard policies, reachable guard nodes. In this subsection we derive results for the simple cases of the no-guards policy and the 1-guard policy, when Tor nodes are always reachable (i.e., there is no churn, so $\varphi = 0$). From Lemmas 1 to 3, we see that we need to calculate $\Pr[X_{\leq\mathsf{t}} = 0]$ for each specific scenario. The next lemma gives the probability of no compromise when a guard node is reused for several circuits.

Lemma 4. *Consider a sequence of ℓ circuits, created with a single guard node and with random exit nodes. Then the probability of no compromise in any of the ℓ circuits is $F_{\gamma,\varepsilon}(\ell) = F(\ell)$ (dropping the subscripts for convenience), where*

$$\Pr[X_{\leq\ell} = 0] = F(\ell) \triangleq 1 - \gamma + \gamma\,(1-\varepsilon)^{\ell}. \tag{11}$$

Proof.

$$\begin{aligned}\Pr[X_{\le \ell} = 0] &= \Pr[X_{\le \ell} = 0 \mid \text{honest g.n.}] \cdot \Pr[\text{honest g.n.}] \\ &\quad + \Pr[X_{\le \ell} = 0 \mid \text{malicious g.n.}] \cdot \Pr[\text{malicious g.n.}] \\ &= 1 \cdot (1-\gamma) + (1-\varepsilon)^{\ell} \cdot \gamma = 1 - \gamma + \gamma(1-\varepsilon)^{\ell}\end{aligned}$$

□

Proposition 1 (Formulas for $\Pr[X_{\le \mathsf{t}} = 0]$ for no guards, single guards). *Let S_{t} be as described. Then for the no guards policy,*

$$\Pr[X_{\le \mathsf{t}} = 0] = (1-\gamma\varepsilon)^{\mathsf{t}}$$

while for a single guard policy,

$$\Pr[X_{\le \mathsf{t}} = 0] = (F(\mathsf{T}))^{\lfloor \mathsf{t}/\mathsf{T} \rfloor} F(\mathsf{t} \bmod \mathsf{T})\,. \tag{12}$$

Proof. In a standard scenario without guard nodes, every circuit setup is independent from the others and, for each user $u \in \mathcal{U}$:

$$\Pr[X_{\le \mathsf{t}}(u) = 0] = (\Pr[X_{\le 1}(u) = 0])^{\mathsf{t}} = (1-\gamma\varepsilon)^{\mathsf{t}}$$

because $\Pr[X_{\le 1}(u) = 0]$ is the probability, for a single circuit, to have at least one honest end node. Then within one period of T, the guard node is fixed, so for $1 \le \mathsf{t} \le \mathsf{T}$, by Lemma 4, $\Pr[X_{\le \mathsf{t}}(u) = 0] = F(\mathsf{t})$. Finally, if $\mathsf{t} = a\mathsf{T} + b$ with $a, b \in \mathbb{N}, b < \mathsf{T}$, we obtain

$$\begin{aligned}\Pr[X_{\le \mathsf{t}}(u) = 0] &= (\Pr[X_{\le \mathsf{T}}(u) = 0])^{a} \cdot F(b) \\ &= F(\mathsf{T})^{a} \cdot F(b)\,.\end{aligned}$$

□

Proposition 2 (Formulas for μ_3 for $\varphi = 0$). *Let S_{t} be as described in Section 4, and let $\mu_3 = \mathbb{E}[\min\{\mathsf{t} | X_{\le \mathsf{t}} > 0\}] = \mathbb{E}[\max\{\mathsf{t} | X_{\le \mathsf{t}} = 0\}] + 1$. Then, without guard nodes,*

$$\mu_3 = \frac{1}{\varepsilon\gamma}. \tag{13}$$

With a single guard node,

$$\mu_3 = \frac{\gamma}{\varepsilon}\left(\frac{\mathsf{T}\varepsilon(1-(1-\varepsilon)^{\mathsf{T}})F(\mathsf{T}) + (1-(\mathsf{T}\varepsilon+1)(1-\varepsilon)^{\mathsf{T}})(1-F(\mathsf{T}))}{(1-F(\mathsf{T}))^2}\right). \tag{14}$$

With more than one guard node and the greedy guard selection policy, $\varphi = 0$ implies that only a single guard node is ever used, so that (14) is valid also for this case. Also, for $\mathsf{T} = 1$, (14) reduces to (13).

Proof. So without guard nodes, (10) becomes

$$\begin{aligned}\mu_3 &= \sum_{\mathsf{t}=1}^{\infty} \mathsf{t} \cdot (1-\gamma\varepsilon)^{\mathsf{t}-1} \cdot \gamma\varepsilon \\ &= \frac{\gamma\varepsilon}{(1-(1-\gamma\varepsilon))^2} = \frac{1}{\gamma\varepsilon}.\end{aligned}$$

With a single guard node and $\mathsf{t} = a\mathsf{T} + b$,

$$\Pr[X_{\leq \mathsf{t}} = 1 \mid X_{\leq \mathsf{t}-1} = 0] = 0 \cdot (1-\gamma) + \varepsilon \frac{\gamma(1-\varepsilon)^{b-1}}{F(b-1)} \ = \varepsilon \frac{\gamma(1-\varepsilon)^{b-1}}{F(b-1)},$$

and (10) becomes

$$\begin{aligned}\mu_3 &= \sum_{a=0}^{\infty} \left(\underbrace{F(\mathsf{T})^a}_{\text{no compromise, } a\mathsf{T} \text{ epochs}} \sum_{b=1}^{\mathsf{T}} \underbrace{(a\mathsf{T}+b)}_{\mathsf{t}} \underbrace{\frac{\varepsilon\gamma(1-\varepsilon)^{b-1}}{F(b-1)}}_{\text{compromised new circuit}} \cdot \underbrace{F(b-1)}_{\text{no compromise, } b-1 \text{ epochs}} \right) \\ &= \varepsilon\gamma \sum_{a=0}^{\infty} \left(\left(1-\gamma+\gamma(1-\varepsilon)^{\mathsf{T}}\right)^a \sum_{b=1}^{\mathsf{T}} (a\mathsf{T}+b)(1-\varepsilon)^{b-1} \right) \\ &= \varepsilon\gamma \sum_{a=0}^{\infty} \left(\left(1-\gamma+\gamma(1-\varepsilon)^{\mathsf{T}}\right)^a \left(\frac{a\mathsf{T}(1-(1-\varepsilon)^{\mathsf{T}})}{\varepsilon} + \frac{(1-(\mathsf{T}(\varepsilon+1)(1-\varepsilon)^{\mathsf{T}})}{\varepsilon^2} \right) \right) \\ &= \varepsilon\gamma \sum_{a=0}^{\infty} \left(\left(1-\gamma+\gamma(1-\varepsilon)^{\mathsf{T}}\right)^a \left(\frac{a\mathsf{T}(1-(1-\varepsilon)^{\mathsf{T}})}{\varepsilon} + \frac{(1-(\mathsf{T}(\varepsilon+1)(1-\varepsilon)^{\mathsf{T}})}{\varepsilon^2} \right) \right) \\ &= \varepsilon\gamma \left(\frac{\mathsf{T}\varepsilon(1-(1-\varepsilon)^{\mathsf{T}})F(\mathsf{T}) + (1-(\mathsf{T}\varepsilon+1)(1-\varepsilon)^{\mathsf{T}})(1-F(\mathsf{T}))}{((1-F(\mathsf{T}))\varepsilon)^2} \right),\end{aligned}$$

which is equivalent to (14). □

Proposition 3 (Formulas for $\mu_4(\mathsf{t})$ for $\varphi = 0$). *In the no-churn case, i.e.* $\varphi = 0$*,*

$$\mu_4(\mathsf{t}) = \begin{cases} \gamma\varepsilon & \text{if } \mathsf{t} \equiv 0 \mod \mathsf{T}, \\ \varepsilon & \text{otherwise.} \end{cases} \tag{15}$$

The general case: $\varphi \geq 0, \mathbf{n} \geq 1, \mathbf{T} \geq 1$. The previous subsection gives an insight into the performance when the churn probability φ is very small. The results for $\varphi = 0$ imply that choosing a *single guard* policy with T as large as possible is the best thing to do, but for practical reasons, the Tor protocol uses guard lists with more than one guard node. Hence we proceed with the case of $\varphi > 0$, with a reset policy as discussed in Section 5.1. In this case, over t epochs,

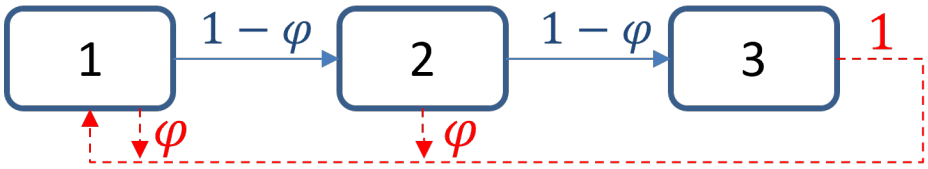


Fig. 3. Paths through Markov chain and their associated probabilities with $\mathbf{n} = 1$ as in Example 2, for the special case of $\mathsf{T} = 3$.

a user will go through a random number m of consecutive guard lists. Since, due to churn, a guard list may be prematurely reset, the i^{th} guard list will actually last for a random number T_i of epochs, where the last guard list may still be active at time t and

$$1 \leq \mathsf{T}_i \leq \mathsf{T}, 1 \leq i \leq m \tag{16}$$

and

$$\sum_{i=1}^{m} \mathsf{T}_i = \mathsf{t}. \tag{17}$$

Thus to calculate the performance of a scenario with a given set of parameters and a given policy, it is necessary to determine the probability distribution of m and the sequence $\mathsf{T}_i, i = 1, \ldots, m$ for each t. We will return to how to calculate $\mu_1(\mathsf{t})$ in Proposition 4, but it is convenient to state a couple of supporting results first. In order to approach the problem of determining our metrics in the $\varphi > 0$ scenario, we start with the simple example of a single guard node.

Example 2. Consider a single guard policy, where the guard node is used for at most T epochs but where the reset policy is to select a new guard (for up to T epochs) if the guard node is unavailable (which occurs with probability φ). The lifetime of the guard node is described by the Markov process shown in Fig. 3, for the case $\mathsf{T} = 3$.

For this example,

$$\begin{aligned} \Pr[X_{\leq \mathsf{t}} = 0] &= \sum_{L=(\mathsf{T}_1,\ldots,\mathsf{T}_{m(L)})\in\mathcal{P}(\mathsf{t})} \Pr[X_{\leq \mathsf{t}} = 0 \mid L] \Pr[L] \\ &= \sum_{L=(\mathsf{T}_1,\ldots,\mathsf{T}_{m(L)})\in\mathcal{P}(\mathsf{t})} \left(\prod_{i=1}^{m(L)} F(\mathsf{T}_i) \right) \Pr[L]. \end{aligned} \tag{18}$$

Here, each T_i represents the number of times the i^{th} guard node has been used since the last renewal of the guard list, and the summation parameter L is a vector $(\mathsf{T}_1, \mathsf{T}_2, \ldots, \mathsf{T}_m)$ that represents a sequence of run lengths. This L runs through the set $\mathcal{P}(\mathsf{t})$ of all ordered integer partitions of t satisfying (16) and (17). A convenient way to compute the sum in (18) is by use of *generating functions* [7].

Consider the functions $G_0(Z) = \sum_{b=1}^{\mathsf{T}} P_b Z^b$ and $G(Z) = \sum_{b=1}^{\mathsf{T}} Q_b Z^b$, where

$$P_b = F(b)\,(1-\varphi)^{b-1} \text{ for } b = 1, \ldots, \mathsf{T}$$

and

$$Q_b = F(b)\,(1-\varphi)^{b-1}\varphi \text{ for } b = 1, \ldots, \mathsf{T}-1$$

while $Q_{\mathsf{T}} = P_{\mathsf{T}} = F(\mathsf{T})\,(1-\varphi)^{\mathsf{T}-1}$. Here P_b and Q_b represent, respectively, the joint probability in the summand of (18) for a path starting in state 1 in the Markov chain (Fig. 3) that has reached state b and then has (resp. has not) returned to the starting state. Then the probability $\Pr[X_{\leq \mathsf{t}} = 0]$ is the coefficient of Z^{t} in the polynomial

$$G_0(Z) \sum_{\ell=0}^{\mathsf{t}'} G(Z)^{\ell}, \tag{19}$$

where $\mathsf{t}' \geq \mathsf{t} - 1$. For the case of $\mathsf{T} = 3$ as in Fig. 3, we have

$$G_0(3) = F(1)\,Z + F(2)\,(1-\varphi)Z^2 + F(3)\,(1-\varphi)^2 Z^3$$

and

$$G(3) = F(1)\,\varphi Z + F(2)\,\varphi(1-\varphi)Z^2 + F(3)\,(1-\varphi)^2 Z^3.$$

Expanding (19) in terms of powers of Z gives

$$\begin{aligned}
&1 + F(1)\,Z + (F(1)^2\,\varphi + F(2)\,(1-\varphi))Z^2 + \\
&(F(3)\,(1-\varphi)^2 + F(1)\,(F(2)\,\varphi(1-\varphi) + F(1)^2\,\varphi^2) + F(2)\,(1-\varphi)F(1)\,\varphi)Z^3 + \\
&(F(1)\,\varphi F(3)\,(1-\varphi)^2 + F(2)\,(1-\varphi)(F(2)\,\varphi(1-\varphi) + F(1)^2\,\varphi^2) + \\
&F(1)\,(F(3)\,(1-\varphi)^2 + 2F(1)\,\varphi^2 F(2)\,(1-\varphi) + F(1)^3\,\varphi^3))Z^4 + \cdots
\end{aligned}$$

and the coefficient of each term Z^{t} is the probability $\Pr[X_{\leq \mathsf{t}} = 0]$, so that

$$\begin{aligned}
\Pr[X_{\leq 0} = 0] &= 1, \\
\Pr[X_{\leq 1} = 0] &= F(1) = 1 - \gamma + \gamma(1-\varepsilon), \\
\Pr[X_{\leq 2} = 0] &= F(1)^2\,\varphi + F(2)\,(1-\varphi) \\
&= (1 - \gamma + \gamma\,(1-\varepsilon))^2\,\varphi + \left(1 - \gamma + \gamma\,(1-\varepsilon)^2\right)(1-\varphi) \\
&= \left(\gamma^2 - \gamma\right)\varepsilon^2\varphi - (2-\varepsilon)\,\gamma\varepsilon + 1,
\end{aligned}$$

and so on. □

Next, we proceed to expand the discussion to include the case where the number of guards in the guard list is $\mathbf{n} > 1$. As noted in Proposition 2, in the case of no churn a *greedy* policy with $\mathbf{n} > 1$ will behave and perform identically to the single guard case, hence we study the $\mathbf{n} > 1$ scenario only for $\varphi > 0$.

For a guard list of $\mathbf{n}$ guards, and for each $i = 1, \ldots, \mathbf{n}$, let ℓ_i be the number of times guard i is used during t epochs, so that $\sum_i \ell_i = \mathsf{t}$. By Lemma 5, γ, ε, and the distribution $\chi = (\ell_1, \ldots, \ell_{\mathbf{n}})$ completely determine $\Pr[X_{\leq \mathsf{t}} = 0]$.

Lemma 5. *Assume that, at time* $\mathtt{t} = \ell_1 + \cdots + \ell_{\mathbf{n}} \leq \mathsf{T}$*, the* $\mathbf{n}$ *different guard nodes from the guard list have been used (in any order) respectively* $\ell_1, \ldots, \ell_{\mathbf{n}}$ *times since the last guard list renewal. Then*

$$\Pr[X_{\leq \mathtt{t}} = 0] = \prod_{i=1}^{\mathbf{n}} F(\ell_i) \tag{20}$$

Proof. Proof is by induction over $\mathbf{n}$, where the base case $\mathbf{n} = 1$ holds by definition. For the induction step:

$$\prod_{i=1}^{\mathbf{n}+1} F(\ell_i) = \left(\prod_{i=1}^{\mathbf{n}} F(\ell_i) \right) \cdot F(\ell_{\mathbf{n}+1}) = \Pr\left[X_{\leq \mathtt{t}-\ell_{\mathbf{n}+1}} = 0\right] \cdot F(\ell_{\mathbf{n}+1}) = \Pr[X_{\leq \mathtt{t}} = 0].$$

□

Lemma 6 (Monotonicity of $F(\ell)$). *Assume, without loss of generality, that* $\ell_i \geq \ell_{i+1}$*, for* $1 \leq i < \mathbf{n}$*. For positive integers* $\ell_1 \geq \ell_2$*,*

$$F(\ell_1 + 1)\, F(\ell_2 - 1) \geq F(\ell_1)\, F(\ell_2) \tag{21}$$

and for positive integers ℓ_1*,* ℓ_2*,*

$$F(\ell_1 + \ell_2) \geq F(\ell_1)\, F(\ell_2)\,. \tag{22}$$

Proof.

$$\begin{aligned} F(\ell_1 + \ell_2) - F(\ell_1)\, F(\ell_2) &= 1 + \gamma + \gamma\,(1-\varepsilon)^{\ell_1+\ell_2} - \left(1 + \gamma + \gamma\,(1-\varepsilon)^{\ell_1}\right) \\ &\quad \cdot \left(1 + \gamma + \gamma\,(1-\varepsilon)^{\ell_2}\right) \\ &= \gamma(1-\gamma)(1 - (1-\varepsilon)^{\ell_1} - (1-\varepsilon)^{\ell_2} + (1-\varepsilon)^{\ell_1+\ell_2}) \\ &= \gamma(1-\gamma)(1 - (1-\varepsilon)^{\ell_1})(1 - (1-\varepsilon)^{\ell_2}) \geq 0, \end{aligned}$$

and, for $\ell_1 \geq \ell_2$,

$$F(\ell_1 + 1)\, F(\ell_2 - 1) - F(\ell_1)\, F(\ell_2) = \frac{(1-\gamma)\gamma\varepsilon((1-\varepsilon)^{\ell_2} - (1-\varepsilon)^{\ell_1+1})}{1-\varepsilon} \geq 0.$$

□

It follows from Lemmas 5 and 6 that a *greedy* policy is optimum with respect to maximizing $\Pr[X_{\leq \ell} = 0]$, and that an *ideally uniform use* algorithm is the worst possible. Hence the performance of any policy (for selecting a guard from the guard list for use for the next circuit) will lie between these two extremes.

Assume that the user has created $\mathtt{t}$ circuits (one circuit per epoch), and that the $\mathbf{n}$ guard nodes have been used respectively $\chi_1, \chi_2, \ldots, \chi_{\mathbf{n}}$ times, where $\chi_1 + \chi_2 + \cdots + \chi_{\mathbf{n}} = \mathtt{t}$ and, without loss of generality, $\chi_1 \geq \chi_2 \geq \cdots \geq \chi_{\mathbf{n}}$. Then let the *guard node distribution* be $\chi(\mathtt{t}) = (\chi_1, \chi_2, \ldots, \chi_{\mathbf{n}})$. The ordering

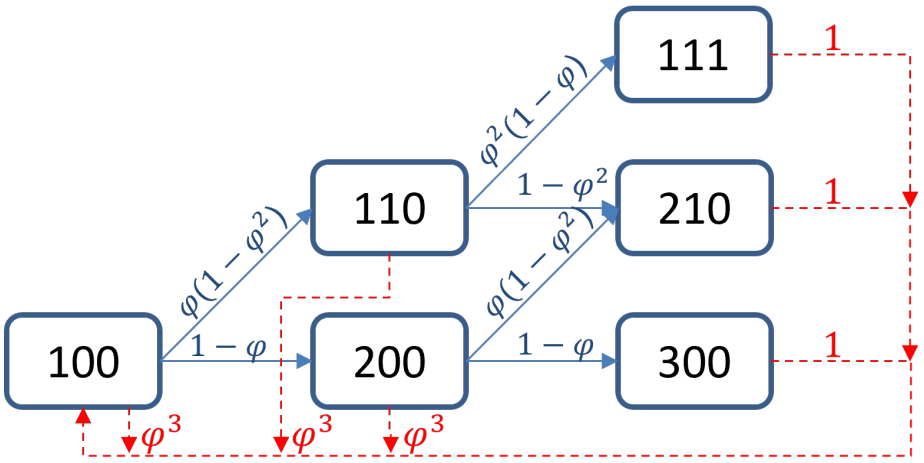


Fig. 4. Paths through Markov chain and their associated probabilities for the greedy policy.

of the actual guard nodes may change over time to maintain the constraint $\chi_1 \geq \chi_2 \geq \cdots \geq \chi_{\mathbf{n}}$. Finally, let $\underline{\chi}(\mathsf{t})$ be the set of all guard distributions at epoch t. At time $\mathsf{t} > 0$, the current guard set has been used for a random time of b epochs, and the Markov process describing the guard use will be in a random state $\chi(b)$.

Let $P_\chi(b)$ be the probability of guard distribution $\chi = (\chi_1, \chi_2, \ldots, \chi_{\mathbf{n}})$ at epoch b, where $b = \chi_1 + \chi_2 + \cdots + \chi_{\mathbf{n}}$ and $1 \leq b \leq \mathsf{T}$. Guard distributions develop over epochs through a random walk if $\varphi > 0$. Let $P_{\chi'(\mathsf{t}-1)\to\chi(\mathsf{t})}$ denote the probability that a guard distribution $\chi'(\mathsf{t}-1)$ at epoch $\mathsf{t}-1$ develops into $\chi(\mathsf{t})$ at epoch t. The probability $P_{\chi'(\mathsf{t}-1)\to\chi(\mathsf{t})}$ depends (only) on φ and the guard selection policy.

For $\chi' = (1, \underbrace{0, \ldots, 0}_{\mathbf{n}-1})$, define $P_{\chi'}(1) = 1$, and for $1 < b \leq \mathsf{T}$ and $\chi \in \underline{\chi}(b)$,

$$P_\chi(b) = \sum_{\chi' \in \underline{\chi}(b-1)} P_{\chi'}(b-1) P_{\chi'(b-1)\to\chi(b)}.$$

Proposition 4 (Formula for $\Pr[X_{\leq \mathsf{t}} = 0]$ for n-guards scenarios). *Let $G_0(P,Z) = \sum_{b=1}^{\mathsf{T}} P_b Z^b$ and $G(Q,Z) = \sum_{b=1}^{\mathsf{T}} Q_b Z^b$, where $P_b = F(b)\, P_\chi(b)$ for $b = 1, \ldots, \mathsf{T}$ and $Q_b = F(b)\, P_\chi(b) \varphi^{\mathbf{n}}$ for $b = 1, \ldots, \mathsf{T}-1$ while $Q_{\mathsf{T}} = F(\mathsf{T})\, P_\chi(\mathsf{T})$. Then for $\forall \mathsf{t} > 1$, $\forall \mathsf{t}' \geq \mathsf{t} - 1$, the probability $\Pr[X_{\leq \mathsf{t}} = 0]$ is the coefficient of Z^{t} in the polynomial*

$$G_0(P,Z) \sum_{\ell=0}^{\mathsf{t}'} G(Q,Z)^\ell. \tag{23}$$

Proof. The generating polynomials $G(Q,Z)$ and $G_0(P,Z)$ represent, respectively, accountancies of completed and uncompleted (current) guard lists, re-

Table 4. Summary of transition probabilities for the greedy policy. The table gives the transition probabilities $P_{\chi' \to \chi}$ for each *type* of guard node distribution at epoch $\mathtt{t}-1$ (column headers) to each corresponding distribution at epoch $\mathtt{t}$ in the *greedy* policy.

$(\chi_1,\chi_2,\chi_3) \to$ at epoch $\mathtt{t}-1$ To state: at epoch $\mathtt{t}$	$\chi_1 > \chi_2 > \chi_3$	$\chi_1 = \chi_2 > \chi_3$	$\chi_1 > \chi_2 = \chi_3$	$\chi_1 = \chi_2 = \chi_3$
(χ_1+1,χ_2,χ_3)	$1-\varphi$	$1-\varphi^2$	$1-\varphi$	$1-\varphi^3$
(χ_1,χ_2+1,χ_3)	$\varphi(1-\varphi)$		$\varphi(1-\varphi^2)$	
(χ_1,χ_2,χ_3+1)	$\varphi^2(1-\varphi)$	$\varphi^2(1-\varphi)$		
RESET	φ^3	φ^3	φ^3	φ^3

Table 5. The table gives the transition probabilities $P_{\chi' \to \chi}$ for each *type* of guard node distribution at epoch $\mathtt{t}-1$ (column headers) to each corresponding distribution at epoch $\mathtt{t}$ in the *IU* policy.

$(\chi_1,\chi_2,\chi_3) \to$ at epoch $t-1$ To state: at epoch $\mathtt{t}$	$\chi_1 > \chi_2 > \chi_3$	$\chi_1 = \chi_2 > \chi_3$	$\chi_1 > \chi_2 = \chi_3$	$\chi_1 = \chi_2 = \chi_3$
(χ_1+1,χ_2,χ_3)	$\varphi^2(1-\varphi)$	$\varphi(1-\varphi)^2$	$\varphi^2(1-\varphi)$	$1-\varphi^3$
(χ_1,χ_2+1,χ_3)	$\varphi(1-\varphi)$		$1-\varphi^2$	
(χ_1,χ_2,χ_3+1)	$1-\varphi$	$1-\varphi$		
RESET	φ^3	φ^3	φ^3	φ^3

spectively. Hence (23) provides a convenient way to add up, for each $\mathtt{t}$, and over all guard node distributions χ of $\mathtt{t}$ epochs (and circuits) the probability of χ times the associated $\Pr[X_{\leq \mathtt{t}} = 0|\chi]$ for that χ. Hence, (23) effectively computes an expectation, namely $\Pr[X_{\leq \mathtt{t}} = 0]$. □

Example 3. Fig. 4 shows the Markov chain for a 3-guards greedy policy, with complete reset of guard list if all nodes are unavailable at time of circuit creation. The transition probabilities $P_{\chi' \to \chi}$ for this scenario are summarised in Table 4. Each column in the table represents a type of guard node distribution, the possible next-epoch distribution, and the probability of the transition as a function of the probability φ of guard unavailability.

Example 4. Fig. 5 shows the Markov chain for a 3-guards IU policy, with complete reset of guard list if all nodes are unavailable at time of circuit creation. The transition probabilities $P_{\chi' \to \chi}$ for this scenario are summarised in Table 5. The model is the same as in Example 3 except that the transition probabilities are different.

5.3 Simulation Program

The analytical approach in Section 5.2 offers expressions that, at least for some metrics, gives "exact" numerical results. However, some expressions are complicated and offer little in terms of intuitive understanding. A simulation process may be easier to apply to most of the metrics we consider, and gives sufficiently precise results. Since *tornettools* does not support the guard node feature, we wrote a program to simulate a Tor scenario simplified as discussed in Section 5.1. To check the correctness of the simulation program, we have verified that the results coincide exactly with our theoretical results where applicable.

For each simulation sample, a set of parameter values is applied to a random process in which guard lists are selected, guards and exit nodes are compromised according to probabilities γ and ε, guard nodes are selected according to a given policy and based on availability according to the churn parameter φ, and a guard list is completely renewed when it has been used for T epochs or all guards are simultaneously unavailable. Each sample is run for a preselectable number of epochs, and data for the metrics are collected for each sample. For each set of parameters, the simulation is run for as many $(10^6 - 10^7)$ samples as needed.

Lemma 7 (Formulas for μ_4 for $\varphi > 0$). *For the case of $\varphi > 0$, the asymptotic value of $\mu_4 = \lim_{\mathsf{t}\to\infty} \mu_4(\mathsf{t})$ is given by*

$$\mu_4 = (\pi_{change}(\gamma - 1) + 1)\varepsilon, \tag{24}$$

where π_{change} is the probability that the guard node used at time $\mathsf{t}+1$ is different from the one used at time t.

Proof (Sketch). Using (7), we condition the probability μ_4 on whether the guard list at time t needs to be renewed due to having reached the full lifetime, or not. □

Remark 4 (Regarding Lemma 7). It can be seen that (with equality for $\mathbf{n} = 1$),

$$\pi_{change} \geq \frac{1}{\sum_{i=1}^{\mathsf{T}} (1 - \varphi^{\mathbf{n}})^i}.$$

For $\mathbf{n} > 1$, π_{change} is a lower bound on π_{change}, since guard changes can occur also due to other reasons. For $\mathbf{n} > 1$, π_{change} can be estimated by simulation. A direct estimator for $\mu_4(\mathsf{t})$ can naïvely be obtained by counting pairs of compromised circuits in a simulation and dividing this number by $\gamma\varepsilon$, but using (24) gives a more precise way of estimating $\mu_4(\mathsf{t})$.

5.4 Discussion: Relevant Parameter Ranges

To provide further insight into the practical ramifications of our work, we next discuss suitable parameter ranges, followed by an interpretation of the results from our analytical model for these parameters.

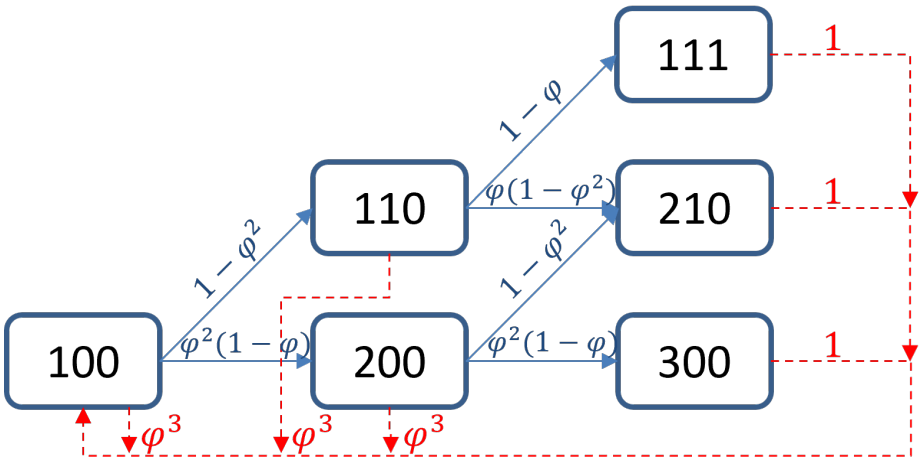


Fig. 5. Paths through Markov chain and their associated probabilities for the even-use policy.

Network characteristics n and φ. Natural churn rate φ in the Tor network can be computed by collecting and comparing subsequent consensus files. We observe from previous research [40, Section 5.2] that φ is typically in the range $[0.001, 0.003]$. For an **n**-guard policy, the probability that all **n** guard nodes are simultaneously unavailable in an epoch is $\varphi^{\mathbf{n}}$. This observation is an argument for using lists with at least three guard nodes. Conversely, it suggests robustness and small downtime as criteria for selecting guard nodes.

Epoch granularity T. In our experiments, we have used both small (to shed light on the mechanisms) and large values of T. Our model is simplified with respect to real users, who will not be continuously connected. However, it seems reasonable that a real user can create some hundred circuits, one epoch per circuit, during a normal Tor guard list lifetime.

Compromise levels γ and ε. We can model different scenarios through careful choice of γ and ε:

- $\varepsilon = 1$: The case when guard compromise is considered equivalent to circuit compromise.
- $0 < \varepsilon < 1$: guard compromise combined with a probability ε of the union of events $\{$*website fingerprinting is successful, exit node is compromised*$\}$.
- $0 < \varepsilon = \gamma < 1$: The case where both ends need to be compromised for a circuit to be compromised; we are concerned with *one single adversary* that controls the fraction $\varepsilon = \gamma$ of Tor node bandwidth.
- $0 < \varepsilon \ll \gamma < 1$: The case where both ends need to be compromised for a circuit to be compromised; we are concerned by compromise by *any* of

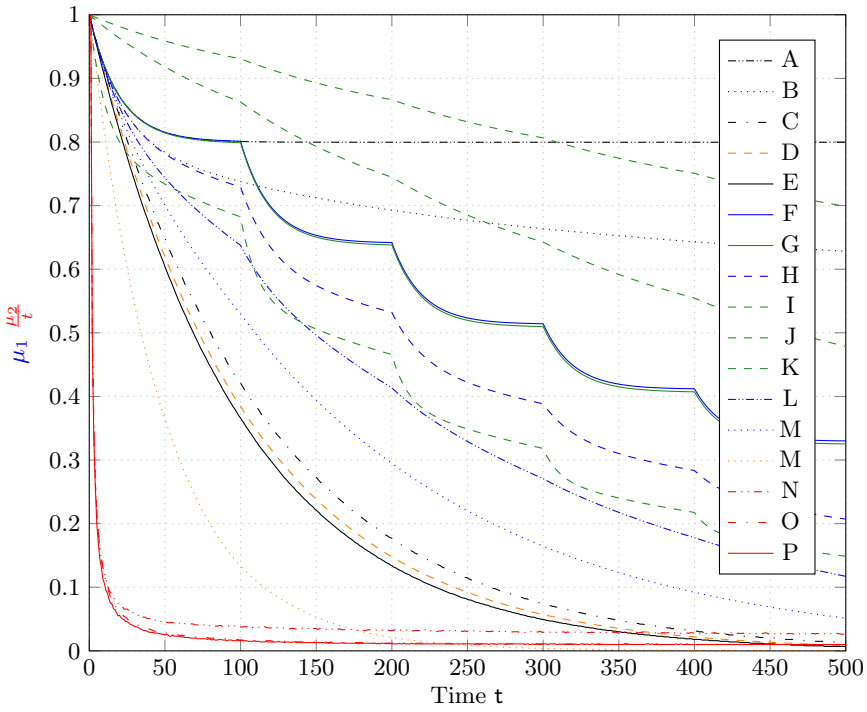


Fig. 6. Simulation results for 500 epochs and parameter sets $(\mathbf{n}, \mathsf{T}, \varphi, \gamma, \varepsilon)$. The parameter sets as listed in the figure legend are A: (1, 1000, 0, 0.2, 0.05), B: (100, 1000, 0.1, 0.2, 0.05), C: (3, 10, 0.1, 0.2, 0.05), D: (3, 10, 0.1, 0.2, 0.05, IU), E: (1, 1, 0, 0.2, 0.05), F: (3, 100, 0), G: (3, 100, 0.003, 0.2, 0.05), H: (3, 100, 0.1, 0.2, 0.05), I: (3, 100, 0.1, 0.2, 0.01), J: (3, 100, 0.1, 0.1, 0.01), K: (3, 100, 0.1, 0.2, 0.1), L: (3, 100, 0.2, 0.2, 0.05), M: (3, 100, 0.3, 0.2, 0.05), N: (100, 1000, 0.1, 0.2, 0.05), O: (3, 10, 0.1, 0.2, 0.05), P: (1, 1, 0, 0.2, 0.05). Unless explicitly stated as IU (ideally uniform), the policy is the greedy one. Black, blue, green, and orange curves show $\mu_1(\mathsf{t})$. The "waves" of the blue and green curves show the effect of changing stable guard sets. The red curves passing through the lower left corner show $\mu_2(\mathsf{t})/t$.

these. Each adversary controls a fraction ε of Tor node bandwidth. Thus, the effective guard node compromise ratio γ is $\varepsilon \times$ (number of adversaries).

5.5 Results

Figure 6 shows results for $\mu_1(\mathsf{t})$ and $\mu_2(\mathsf{t})$ for selected sets of parameters. From the results, different patterns emerge:

- The value of $\mu_1(\mathsf{t})$ for the greedy policy quickly converges to its expected value of $(1-\gamma)$, until the guard list is renewed.
- The value of $\mu_1(\mathsf{t})$ for the IU policy is sometimes close to the greedy policy (curves C and D), and always lower bounded by the no-guard policy (curve E). For the parameter set corresponding to curve B, the IU policy curve (not

explicitly shown) coincides almost exactly with curve E. In general, for large $\mathbf{n}$ and T and small φ, the guard selection policy influences $\mu_1(\mathtt{t})$ heavily.
- The value of $\mu_2(\mathtt{t})$ for the greedy policy quickly converges to (9).
- For the realistic $\varphi \in [0, 0.003]$ there is no significant impact of φ on any of our metrics. Thus our results for all four metrics conditioned on $\varphi = 0$ are good approximations.
- In general, the two other metrics (not shown due to lack of space) seem to vary less dramatically.

6 Conclusion

We have presented a general evaluation framework to evaluate the security performance of Tor network protocol policies and the efficacy of attacks carried out by adversaries observing and possibly interacting with the Tor network over time. As an example, we have used the framework to develop an analysis of the guard node feature, shedding new insights on the guard node selection as distinct from the guard list maintenance and the effects that various parameters have on the examined metrics.

References

1. Backes, M., Goldberg, I., Kate, A., Mohammadi, E.: Provably secure and practical onion routing. In: Zdancewic, S., Cortier, V. (eds.) CSF 2012 Computer Security Foundations Symposium. pp. 369–385. IEEE Computer Society Press (2012). https://doi.org/10.1109/CSF.2012.32
2. Backes, M., Kate, A., Manoharan, P., Meiser, S., Mohammadi, E.: AnoA: A framework for analyzing anonymous communication protocols. In: Cortier, V., Datta, A. (eds.) CSF 2013 Computer Security Foundations Symposium. pp. 163–178. IEEE Computer Society Press (2013). https://doi.org/10.1109/CSF.2013.18
3. Backes, M., Kate, A., Meiser, S., Mohammadi, E.: (Nothing else) MATor(s): Monitoring the anonymity of tor's path selection. In: Ahn, G.J., Yung, M., Li, N. (eds.) ACM CCS 2014. pp. 513–524. ACM Press (Nov 2014). https://doi.org/10.1145/2660267.2660371
4. Backes, M., Manoharan, P., Mohammadi, E.: TUC: Time-sensitive and modular analysis of anonymous communication. In: Datta, A., Fournet, C. (eds.) CSF 2014 Computer Security Foundations Symposium. pp. 383–397. IEEE Computer Society Press (2014). https://doi.org/10.1109/CSF.2014.34
5. Boda, K., Földes, Á.M., Gulyás, G.G., Imre, S.: User tracking on the web via cross-browser fingerprinting. In: Laud, P. (ed.) Information Security Technology for Applications. Lecture Notes in Computer Science, vol. 7161, pp. 31–46. LNCS, Berlin, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29615-4_4
6. Cover, T.M., Thomas, J.A.: Elements of Information Theory. John Wiley & Sons, Ltd, USA (2006). https://doi.org/10.1002/047174882X
7. Cox, D., Miller, H.: The Theory of Stochastic Processes. Chapman & Hall (1965). https://doi.org/10.1201/9780203719152

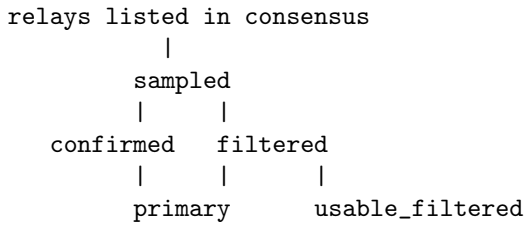
8. Das, A., Acar, G., Borisov, N., Pradeep, A.: The web's sixth sense: A study of scripts accessing smartphone sensors. In: Lie, D., Mannan, M., Backes, M., Wang, X. (eds.) ACM CCS 2018. pp. 1515–1532. ACM Press (Oct 2018). https://doi.org/10.1145/3243734.3243860
9. Degabriele, J.P., Stam, M.: Untagging Tor: A formal treatment of onion encryption. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part III. LNCS, vol. 10822, pp. 259–293. Springer, Heidelberg (Apr / May 2018). https://doi.org/10.1007/978-3-319-78372-7_9
10. Dingledine, R., Hopper, N., Kadianakis, G., Mathewson, N.: One fast guard for life (or 9 months) (2014)
11. Dingledine, R., Mathewson, N.: Tor path specification (Dec 2021), https://github.com/torproject/torspec/blob/main/path-spec.txt, commit 0911bbd
12. Dingledine, R., Mathewson, N.: Tor protocol specification (Nov 2021), https://github.com/torproject/torspec/blob/main/tor-spec.txt, commit 48ab890
13. Dingledine, R., Mathewson, N., Syverson, P.F.: Tor: The second-generation onion router. In: Blaze, M. (ed.) USENIX Security 2004. pp. 303–320. USENIX Association (Aug 2004)
14. Elahi, T., Bauer, K.S., AlSabah, M., Dingledine, R., Goldberg, I.: Changing of the guards: A framework for understanding and improving entry guard selection in Tor. In: Yu, T., Borisov, N. (eds.) WPES 2012. pp. 43–54. ACM (Oct 2012). https://doi.org/10.1145/2381966.2381973
15. Englehardt, S., Narayanan, A.: Online tracking: A 1-million-site measurement and analysis. In: Weippl, E.R., Katzenbeisser, S., Kruegel, C., Myers, A.C., Halevi, S. (eds.) ACM CCS 2016. pp. 1388–1401. ACM Press (Oct 2016). https://doi.org/10.1145/2976749.2978313
16. Fu, X., Ling, Z.: One cell is enough to break Tor's anonymity (2009)
17. Goldschlag, D.M., Reed, M.G., Syverson, P.F.: Hiding routing information. In: Anderson, R.J. (ed.) IWIH 1996. LNCS, vol. 1174, pp. 137–150. Springer, Heidelberg, Berlin, Heidelberg (Jun 1996). https://doi.org/10.1007/3-540-61996-8_37
18. Goldschlag, D.M., Reed, M.G., Syverson, P.F.: Onion routing. Communications of the Association for Computing Machinery **42**(2), 39–41 (Feb 1999). https://doi.org/10.1145/293411.293443
19. Hanley, H., Sun, Y., Wagh, S., Mittal, P.: DPSelect: A differential privacy based guard relay selection algorithm for tor. PoPETs **2019**(2), 166–186 (Apr 2019). https://doi.org/10.2478/popets-2019-0025
20. Hayes, J., Danezis, G.: Guard sets for onion routing. PoPETs **2015**(2), 65–80 (Apr 2015). https://doi.org/10.1515/popets-2015-0017
21. Hevia, A., Micciancio, D.: An indistinguishability-based characterization of anonymous channels. In: Borisov, N., Goldberg, I. (eds.) PETS 2008. LNCS, vol. 5134, pp. 24–43. Springer, Heidelberg (Jul 2008). https://doi.org/10.1007/978-3-540-70630-4_3
22. Jansen, R., Hopper, N.: Shadow: Running Tor in a box for accurate and efficient experimentation. In: NDSS 2012. The Internet Society (Feb 2012)
23. Jansen, R., Tracey, J., Goldberg, I.: Once is never enough: Foundations for sound statistical inference in tor network experimentation. In: Bailey, M., Greenstadt, R. (eds.) USENIX Security 2021. pp. 3415–3432. USENIX Association (Aug 2021)
24. Johnson, A., Wacek, C., Jansen, R., Sherr, M., Syverson, P.F.: Users get routed: traffic correlation on Tor by realistic adversaries. In: Sadeghi, A.R., Gligor, V.D., Yung, M. (eds.) ACM CCS 2013. pp. 337–348. ACM Press (Nov 2013). https://doi.org/10.1145/2508859.2516651

25. Karunanayake, I., Ahmed, N., Malaney, R., Islam, R., Jha, S.K.: De-anonymisation attacks on Tor: A survey. IEEE Communications Surveys and Tutorials **23**(4), 2324–2350 (Jul 2021). https://doi.org/10.1109/COMST.2021.3093615
26. Köster, K., Marx, M., Kunstmann, A., Federrath, H.: Evaluation of circuit lifetimes in Tor. In: Meng, W., Fischer-Hübner, S., Jensen, C.D. (eds.) SEC. IFIP Advances in Information and Communication Technology, vol. 648, pp. 142–157. Springer, Heidelberg, Cham (2022)
27. Kuhn, C., Beck, M., Schiffner, S., Jorswieck, E.A., Strufe, T.: On privacy notions in anonymous communication. PoPETs **2019**(2), 105–125 (Apr 2019). https://doi.org/10.2478/popets-2019-0022
28. Kuhn, C., Beck, M., Strufe, T.: Breaking and (partially) fixing provably secure onion routing. In: 2020 IEEE Symposium on Security and Privacy. pp. 168–185. IEEE Computer Society Press (May 2020). https://doi.org/10.1109/SP40000.2020.00039
29. Libert, T.: Exposing the hidden web: An analysis of third-party HTTP requests on 1 million websites (2015), https://arxiv.org/abs/1511.00619
30. Lovecruft, I., Kadianakis, G., Bini, O., Mathewson, N.: Another algorithm for guard selection (Aug 2017), https://github.com/torproject/torspec/blob/main/proposals/271-another-guard-selection.txt, commit db17344
31. Lovecruft, I., Kadianakis, G., Bini, O., Mathewson, N.: Tor guard specification (Oct 2021), https://github.com/torproject/torspec/blob/main/guard-spec.txt, commit 29245fd
32. Lu, T., Du, Z., Wang, Z.J.: A survey on measuring anonymity in anonymous communication systems. IEEE Access **7**, 70584–70609 (May 2019). https://doi.org/10.1109/ACCESS.2019.2919322
33. Mathewson, N., Murdoch, S.: Top changes in Tor since the 2004 design paper (part 2) (Oct 2012), https://blog.torproject.org/top-changes-tor-2004-design-paper-part-2/
34. Mauw, S., Verschuren, J., de Vink, E.P.: A formalization of anonymity and onion routing. In: Samarati, P., Ryan, P.Y.A., Gollmann, D., Molva, R. (eds.) ESORICS 2004. LNCS, vol. 3193, pp. 109–124. Springer, Heidelberg (Sep 2004). https://doi.org/10.1007/978-3-540-30108-0_7
35. Melloni, A., Stam, M., Ytrehus, Ø.: On evaluating anonymity of onion routing. In: AlTawy, R., Hülsing, A. (eds.) SAC 2021. LNCS, vol. 13203, pp. 3–24. Springer, Heidelberg (Sep 2021). https://doi.org/10.1007/978-3-030-99277-4_1
36. Mohd Aminuddin, M.A.I., Zaaba, Z.F., Samsudin, A., Zaki, F., Anuar, N.B.: The rise of website fingerprinting on Tor: Analysis on techniques and assumptions. Journal of Network and Computer Applications **212**(C), 103582 (Mar 2023). https://doi.org/10.1016/j.jnca.2023.103582
37. Øverlier, L., Syverson, P.: Locating hidden servers. In: 2006 IEEE Symposium on Security and Privacy. pp. 100–114. IEEE Computer Society Press (May 2006). https://doi.org/10.1109/SP.2006.24
38. Pries, R., Yu, W., Fu, X., Zhao, W.: A new replay attack against anonymous communication networks. In: ICC 2008. pp. 1578–1582. IEEE (May 2008). https://doi.org/10.1109/ICC.2008.305
39. Reed, M.G., Syverson, P.F., Goldschlag, D.M.: Proxies for anonymous routing. In: ACSAC 1996. pp. 95–104. IEEE Computer Society (1996). https://doi.org/10.1109/CSAC.1996.569678
40. Sharma, P.K., Chaudhary, S., Hassija, N., Maity, M., Chakravarty, S.: The road not taken: Re-thinking the feasibility of voice calling over tor. PoPETs **2020**(4), 69–88 (Oct 2020). https://doi.org/10.2478/popets-2020-0063

41. Star Trek II: The Wrath of Khan (1982), Paramount Pictures, directed by Nicholas Meyer
42. Sun, Y., Edmundson, A., Feamster, N., Chiang, M., Mittal, P.: Counter-RAPTOR: Safeguarding tor against active routing attacks. In: 2017 IEEE Symposium on Security and Privacy. pp. 977–992. IEEE Computer Society Press (May 2017). https://doi.org/10.1109/SP.2017.34
43. Syverson, P.F., Goldschlag, D.M., Reed, M.G.: Anonymous connections and onion routing. In: 1997 IEEE Symposium on Security and Privacy. pp. 44–54. IEEE Computer Society Press (1997). https://doi.org/10.1109/SECPRI.1997.601314
44. The23rd Raccoon: Analysis of the relative severity of tagging attacks (2012), https://lists.torproject.org/pipermail/tor-dev/2012-March/003347.html
45. Wright, M.K., Adler, M., Levine, B.N., Shields, C.: Defending anonymous communications against passive logging attack. In: 2003 IEEE Symposium on Security and Privacy. pp. 28–43. IEEE Computer Society Press (May 2003). https://doi.org/10.1109/SECPRI.2003.1199325

A Tor Guard Specification

The current Tor guard node specification [31] is based on a series of subsequent sampling performed by the proxies, starting from the set of all the current guard nodes according to some probability distribution and some further processing. The specification contains a scheme of the process leading to guard node selection:



We simplify some of the technicalities of the process, while maintaining enough details for a meaningful representation:

1. the guard nodes information is collected from the consensus.
2. 60 guard nodes are `sampled`. This list is persistent across runs of the proxy and ordered by priority. Nodes are removed from this list when they are either out of the consensus long enough or too "old" (expired lifetime).
3. 20 guards are selected each run (not persistent) to be `filtered` according to path selection bias and other settings (operational configuration). Reachable nodes are set `usable_filtered`.
4. `confirmed` guards are the ones the proxy used in the past, as persistent ordered list. Nodes removed from the `sampled` list are removed from this list as well.
5. The top 3 nodes in the intersection of `confirmed` and `filtered` are the `primary` guards. Since `filtered` is not persistent, neither is this list. When building a circuit, the proxy picks the reachable one with highest priority.