Poster: Quantifying Information Flow for Dynamic Secrets

Piotr Mardziel,[†] Mário S. Alvim,[+] Michael Hicks,[‡] and Michael R. Clarkson[*]
([†]student,[‡]faculty) University of Maryland, College Park
[+](faculty) Universidade Federal de Minas Gerais
[*](faculty) George Washington University

I. Overview

Quantitative information-flow models and analyses typically assume that secret information is *static*. But real-world secrets evolve over time. Passwords, for example, should be changed periodically. Cryptographic keys have periods after which they must be retired. Memory offsets in address space randomization techniques are periodically regenerated. Medical diagnoses evolve, military convoys move, and mobile phones travel with their owners. Leaking the current value of these secrets is undesirable. But if information leaks about how these secrets change, adversaries might also be able to predict future secrets or infer past secrets. For example, an adversary who learns how people choose their passwords might have an advantage in guessing future passwords. Similarly, an adversary who learns a trajectory can infer future locations. So it is not just the current value of a secret that matters, but also how the secret changes. Methods for quantifying leakage and protecting secrets should, therefore, account for these *dynamics*.

This work initiates the study of quantitative information flow (henceforth, QIF) for dynamic secrets. First, we present a core model of programs that compute with dynamic secrets. We use *probabilistic automata* to model program execution. These automata are interactive: they accept inputs and produce outputs throughout execution. The output they produce is a random function of the inputs. To capture the dynamics of secrets, our model uses *strategy functions* to generate new inputs based on the history of inputs and outputs. For example, a strategy function might yield the GPS coordinates of a high-security user as a function of time, and of the path the user has taken so far.

Our model includes *wait-adaptive adversaries*, which are adversaries that can observe execution of a system, waiting until a point in time at which it appears profitable to attack. For example, an attacker might delay attacking until collecting enough observations of a GPS location to reach a high confidence level about that location. Or an attacker might passively observe application outputs to determine memory layout, and once determined, inject shell code that accesses some secret.

Second, we propose an information-theoretic metric for quantifying flow of dynamic secrets. Our metric can be used to quantify leakage of the current value of the secret, of a secret at a particular point in time, of the history of secrets, or even of the strategy function that produces the secrets. We show how to construct an optimal wait-adaptive adversary with respect to the metric, and how to quantify that adversary's expected *gain*, as determined by a scenario-specific *gain function*. These functions consider when, as a result of an attack, the adversary might learn all, some, or no information about dynamic secrets. We show that our metric generalizes previous metrics for quantifying leakage of static secrets, including vulnerability, guessing entropy, and g-vulnerability. We also show how to limit the power of the adversary, such that it cannot influence inputs or delay attacks.

Finally, we put our model and metric to use by implementing them in a probabilistic programming language and conducting a series of experiments. Several conclusions can be drawn from these experiments:

- Frequent change of a secret can increase leakage, even though intuition might initially suggest that frequent changes should decrease it. The increase occurs when there is an underlying order that can be inferred and used to guess future (or past) secrets.
- Wait-adaptive adversaries can derive significantly more gain than adversaries who cannot adaptively choose when to attack. So ignoring the adversary's adaptivity (as in prior work on static secrets) might lead one to conclude secrets are safe when they really are not.
- A wait-adaptive adversary's expected gain increases monotonically with time, whereas a non-adaptive adversary's gain might not.
- Adversaries that are *low adaptive*, meaning they are capable of influencing their observations by providing low-security inputs, can learn exponentially more information than adversaries who cannot provide inputs.

Details about our experiments, our model, its relationship to prior information approaches, and our implementation are discussed in the full paper [1]. The extended version of the work, published as a technical report [2], also includes a discussion and the measurement of information flow relative to a memory-limited adversaries and proofs of the various claims of the conference paper. In the rest of this extended abstract we briefly summarize the some of the experiments that demonstrate the above conclusions.

II. Experiments

Our experiments analyze several examples modeled by the following basic scenario. Suppose an illicit-substance dealer is locked in an ever-persistent game of hiding his stash from the police. The simplest form of this example resembles password

guessing, replacing the password with the location of the stash and authentication attempts with "stakeouts" in which police observe a potential stash location for the presence of the stash. After making observations the police will have a chance to "raid" the stash, potentially succeeding. In the meantime the stash location might change.

We consider several variations of this basic scenario, and study how the variations affect the quantity of revealed information. In particular, we consider (1) the impact of low-adaptivity on information leakage (i.e., how the adversary's ability to choose how to influence the system based on observing its outputs affects how much information is leaked), (2) the impact of wait-adaptivity (i.e., how the adversary's ability to wait for the best moment to attack changes the expected information leakage), and (3) how and whether a secret changes impacts the information leaked.

Low adaptivity. The power of the adversary to adaptively provide inputs to the system based on the results of prior observations has significant impact on the vulnerability of a (non-moving) secret. In this experiment we modeled a hidden stash's location as a value drawn uniformly between 0 and 7. The police observe whether the stash is east or west (greater or lesser) than their chosen stakeout location (which is one of eight possibilities). Any fixed ordering of the stakeout locations results in complete knowledge of the secret. On the other hand, adaptively, the police can perform binary search and determine the stash in 3 stakeouts. This is demonstrated in Figure 1 which plots the expected chance of a successful police raid (termed their *gain*) after a varying number of stakeouts. The wide gray lines represent all possible stakeout schedules of which one is highlighted. The thin dark line is the adversary gain given adaptive stakeout locations. The different between the two is exponential.

Wait adaptivity. Wait adaptivity is the ability of the adversary to determine adaptively when they will exploit a system given their state of knowledge; they might choose to wait if they expect to learn significantly more in the future. In our next experiment, we have the same situation of a hidden stash among 8 values but now it is randomly moved to one the 8 locations change every 4 time steps. Additionally the police stakeouts only learn whether the stash was at the exact location of the stakeout (not whether the stash might be nearby, as in the first experiment). In such examples the locations of the stakeouts do not matter as long as they are distinct. The order used here is sequential, wrapping around every 8 time steps. The resulting expected chance of a successful raid after a varying number of stakeouts is shown in Figure 2. The gray line shows how much gain the police would derive were they to choose the time of the raid before making observations. It is seen there that their chances drop to 1 in 8 every time the stash is relocated. On the other hand if the police are adaptively deciding when to raid, their expected success increases monotonically with time. The optimal behavior of the police is to raid only when they successfully observe the stash but otherwise continue observing. The monotonic increase in chances of success is a general property of any scenario with a wait-adaptive adversary.

Frequent change =? good. It is intuitive to think that it is better to change the secret more often than less as in the previous example. This is not always the case. In this

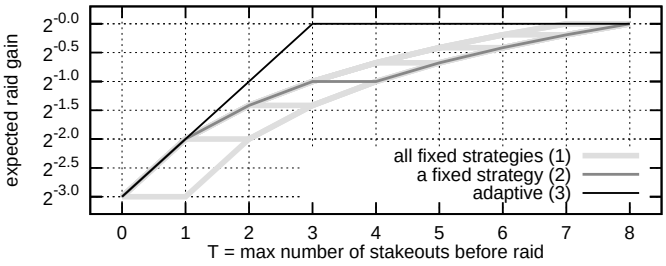

Fig. 1. Non-adaptive and low-adaptive adversary gain.

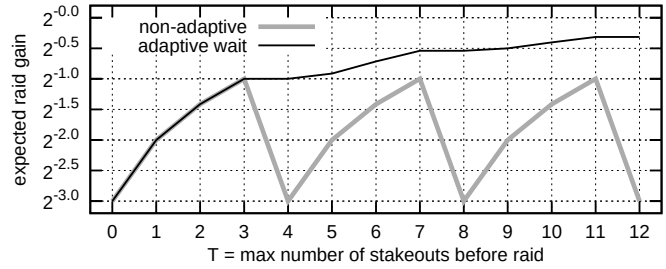

Fig. 2. Non-adaptive and wait-adaptive adversary gain.

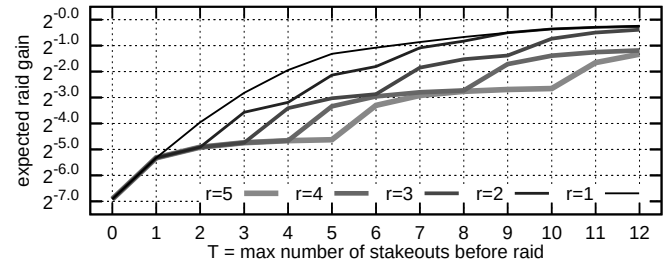

Fig. 3. Impact of varying change frequency.

experiment we modeled a situation where the stash changes in one of many possible deterministic patterns (where a pattern is a simply a permutation). The secret stash location is encoded as a tuple, half of which is the identity of the movement pattern which is never directly observed whereas the other half is a value that is observed directly by the police 9 out of 10 times they make a stakeout. To precisely pinpoint the stash, the police need to know both parts of the secret and they can only learn the first by observing how the latter changes. As such, the more often the secret evolves, the quicker the police will learn permutation, and thus the (upcoming) location of the stash. This is summarized in Figure 3 for varying rates of stash change (the parameter r is the frequency of change).

References

[1] P. Mardziel, M. S. Alvim, M. Hicks, and M. Clarkson, "Quantifying information flow for dynamic secrets," in *Proceedings of the IEEE Symposium on Security and Privacy (Oakland)*, May 2014, to appear.

[2] P. Mardziel, M. S. Alvim, M. Hicks, and M. R. Clarkson, "Quantifying information flow for dynamic secrets," University of Maryland Department of Computer Science, Tech. Rep. CS-TR-5035, 2014, (extended technical report).