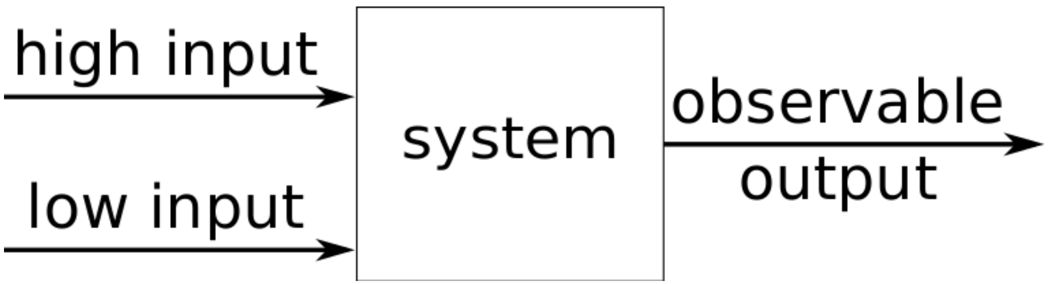
QUANTIFYING INFORMATION FLOW FOR DYNAMIC SECRETS

Piotr Mardziel (UMD, USA)
Mário S. Alvim (UFMG, Brazil)
Michael Hicks (UMD, USA)
Michael R. Clarkson (Cornell, USA)

Quantitative Information Flow Day
PRINCESS workshop 16.Dec.2014

Traditional approach to QIF

- A **system** (probabilistically) maps:
  - a (secret) **high input**, and
  - a (public, adversarially-controlled) **low-input**
  - to a (publically) **observable output**.



- **Leakage** is defined as:

$$leakage = initial\ uncertainty\ - remaining\ uncertainty$$

- Mathematically, given a **measure of uncertainty** F:

$$leakage = F(H) - \sum_{o} p(o) F(H \mid L = \ell, O = o)$$

Why dynamic secrets?

- Traditional quantitative information flow (QIF) models and analyses typically assume that secrets are static.

- But real secrets may evolve over time:
  - Crypto keys must be refreshed after a certain period;
  - Memory offsets in address space randomization techniques are periodically regenerated;
  - Medical diagnoses evolve.

- The *current value* of a secret is sensitive information, but learning *how secrets change* might allow the adversary to infer past or future secrets:
  - Password generation strategies;
  - Learning a trajectory may imply learning future and past locations.

This talk in a nutshell

- We propose a model to represent:
  - Probabilistic, interactive systems,
  - in the presence of adaptive adversaries, and
  - dynamic secrets.

- We show how to quantify the leakage of:
  - The current value of a secret;
  - The value of a secret in any point in time (past of future);
  - The history of secrets;
  - The strategy according to which secrets change.

- The metrics are based on gain-functions [Alvim, Chatzikokolakis, Palamidessi, and Smith, CSF'12]

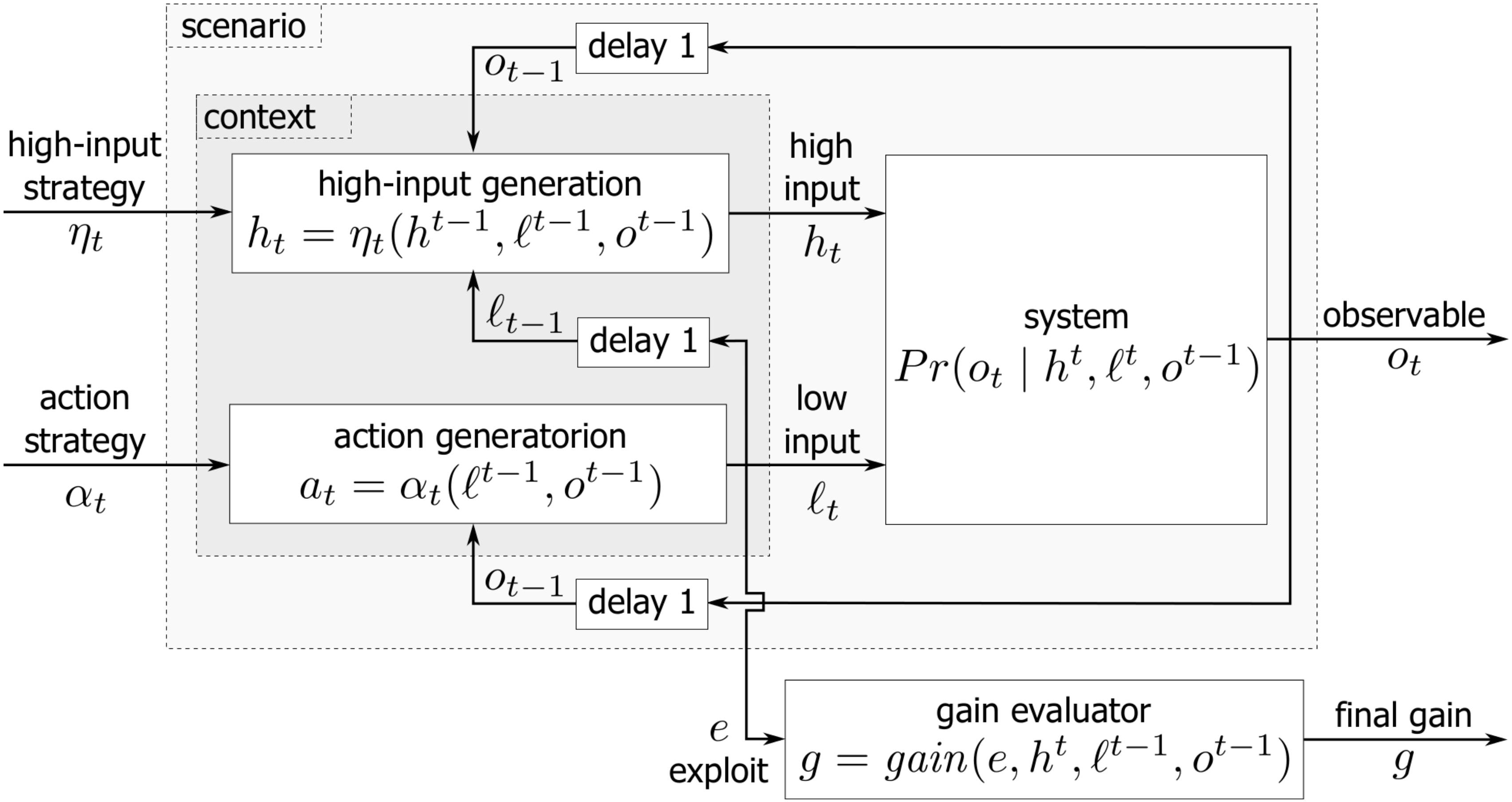
This talk in a nutshell

- Experiments implemented in a probabilistic programming language show that:
  - Adversaries allowed to wait to perform an attack lead to more leakage;
  - Wait-adaptivity always increases gain monotonically, whereas non-ataptive wait may not.
  - Refreshing a secret too often may increase leakage **(!)**

Towards QIF for dynamic secrets

- We extend the traditional model for QIF to encompass:
  - **Interactivity**:
    - multiple uses of the system;
    - the output at a time may influence the input of a subsequent time;
  - **Distinguishing between input and attack**:
    - classically, a system has a single low input;
    - we consider that some inputs may not be attacks;
      - Ex: an adversary navigating a website before launching a SQL injection attack;
    - our model supports quantifying leakage only when attacks occur;
  - **Wait adaptivity**: combining the two features above, adversaries can choose when to attack based on the interaction with the system;
  - **Moving target**: new secrets potentially replace old secrets.

The model


scenario
delay 1
o_{t-1}
context
high-input strategy
η_t
high-input generation
$h_t = \eta_t(h^{t-1}, \ell^{t-1}, o^{t-1})$
high input
h_t
ℓ_{t-1}
delay 1
system
$Pr(o_t \mid h^t, \ell^t, o^{t-1})$
observable
o_t
action strategy
α_t
action generatorion
$a_t = \alpha_t(\ell^{t-1}, o^{t-1})$
low input
ℓ_t
o_{t-1}
delay 1
e
exploit
gain evaluator
$g = gain(e, h^t, \ell^{t-1}, o^{t-1})$
final gain
g

An example: password checker

- High-input: real password
- Low-input: adversary's guess
- Observables: $\{accept, reject\}$
- System:

$$\forall_{1 \le t \le T}: \Pr(o_t = accept) = 1 \text{ iff } h_t = \ell_t$$
$$\forall_{1 \le t \le T}: \Pr(o_t = reject) = 1 \text{ iff } h_t \neq \ell_t$$

- Exploit: choose as attack the guessed password.

- High-input strategy: a new password cannot be the same as the 10 more common guesses, or the last 5 used passwords.
  - It depends on the history of high and low inputs and of observables.
- Action-strategy: an adversary will not try the same guess again until it is likely that the secret has changed.
  - It depends on the history of low inputs and of observables.

Quantifying leakage

- Given a model m and a gain function g, the **dynamic gain** of a scenario is given by:

$$D_g = \max_{s \in Action\ Strategies} E[m, g, s]$$

- The model allows for the quantification of leakage for:
  - Moving target
  - Specific past gain
  - Historical gain
  - Change inference

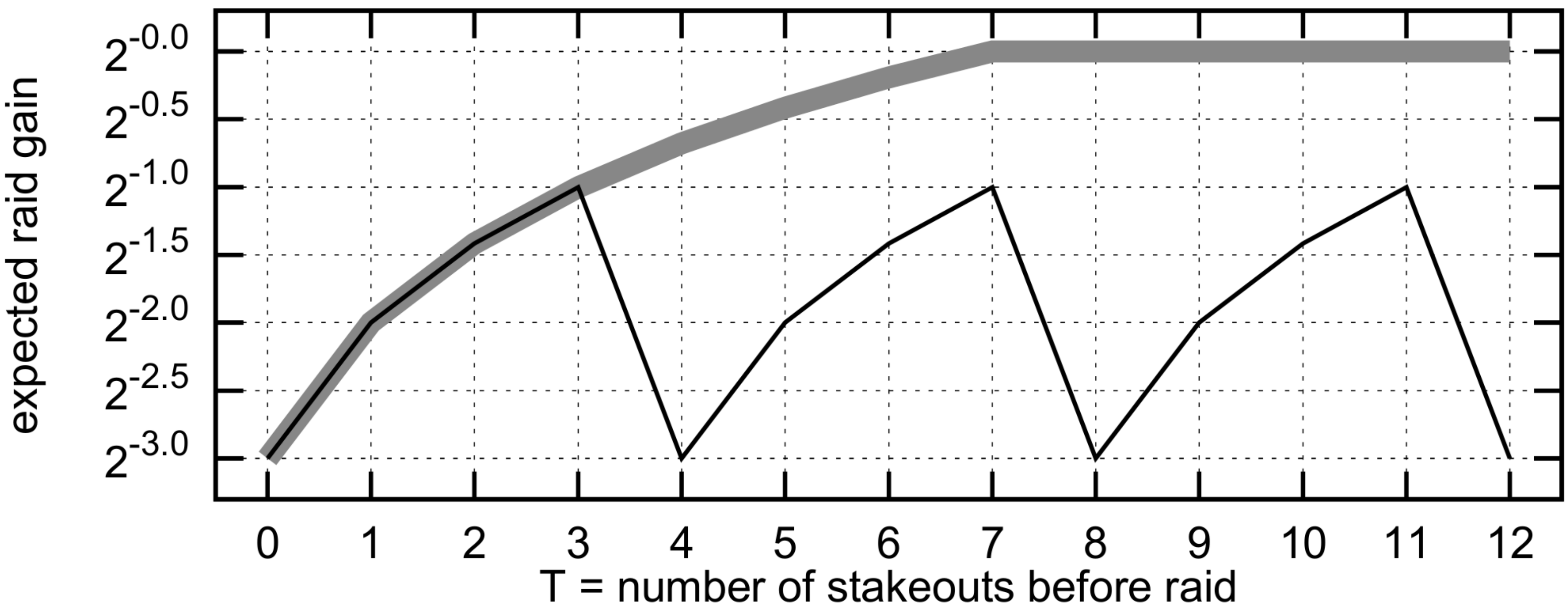
Implementation and experiments

- The model was implemented in a probabilistic programming language based on OCaml.

- Experiments: Stakeouts and raids
  - An illicit stash is hidden in one of several possible locations $\{0,1,\ldots,7\}$ (**high-input**);
  - The police can stakeout and observe suspicious movements outside a location (**low-input**);
  - When certain enough, the police raids a location: the police's gain is 1 when the illicit stash is apprehended, and 0 otherwise (**exploit and gain evaluation**).
  - A gang randomly picks a new location for stash every 4 time steps (**high-input strategy**).

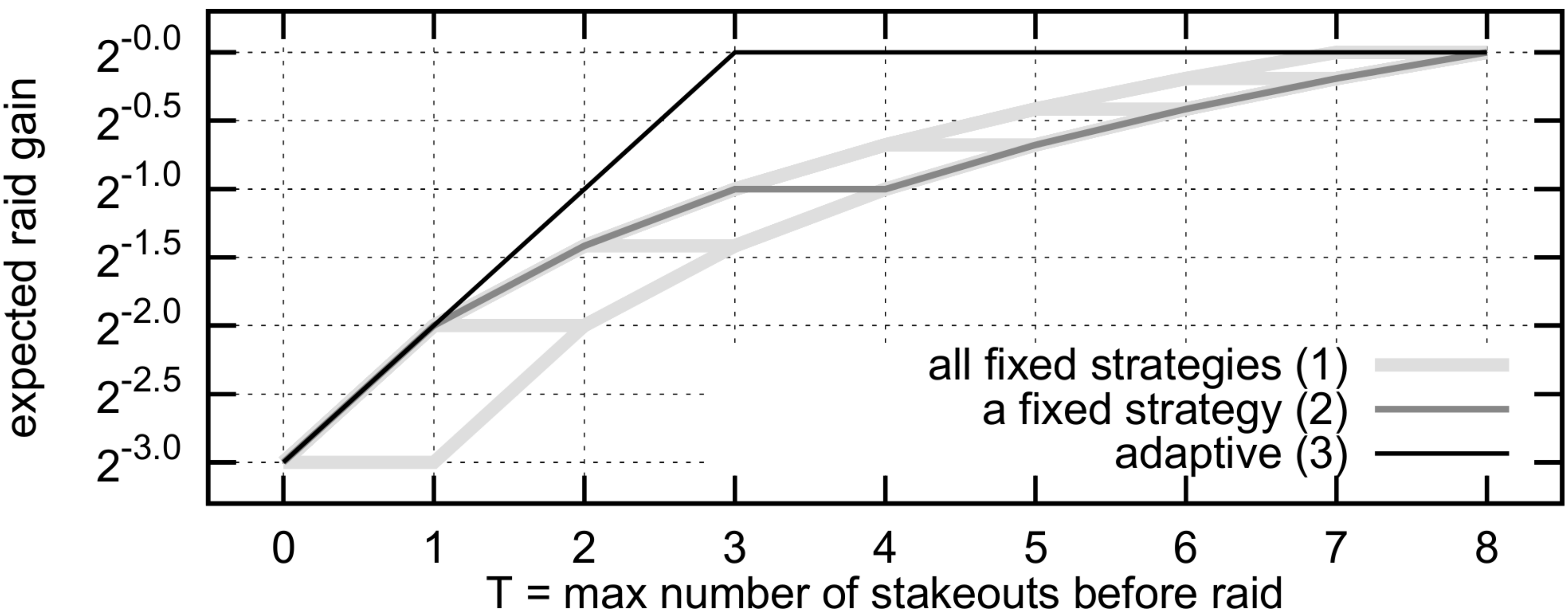
A: Dynamic vs. static secrets

- In general, refreshing a secret limits the information leakage.



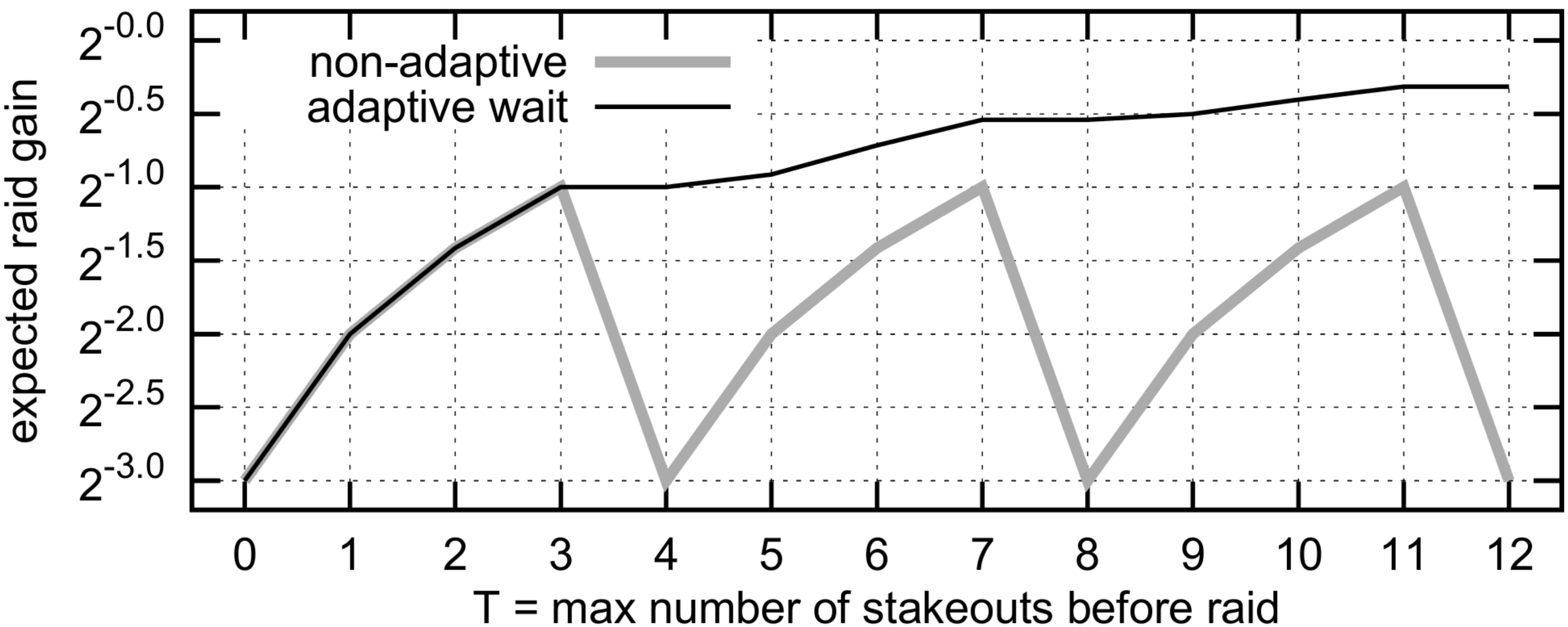
B: Low-adaptivity

- In general, a low-adaptive adversary learns exponentially more information than a non-adaptive one.



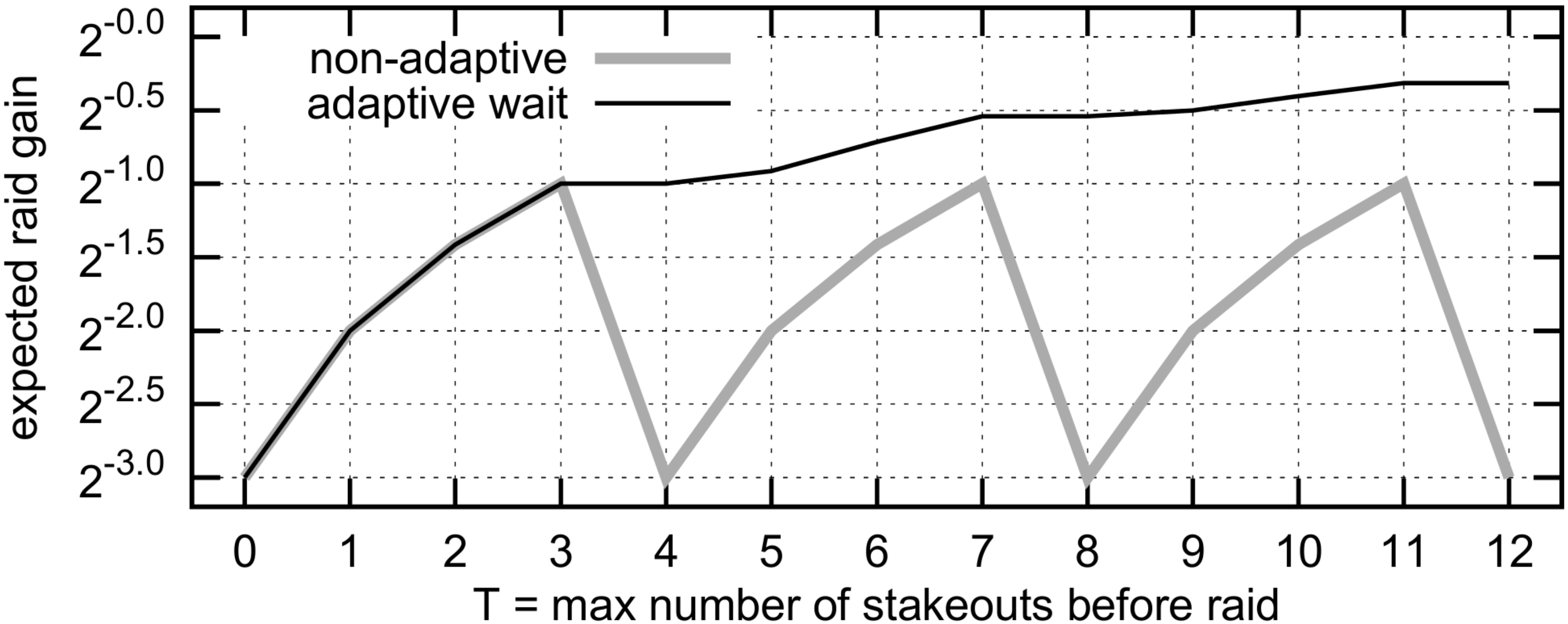
C: Wait-adaptivity

- Intuitively, an optimal wait-adaptive adversary waits until a successful stakeout before attacking.
  - The more observations there are, the more likely this will occur.



C: Wait-adaptivity

- Adversary has to attack before $t = 5$ and has not yet observed a successful stakeout:
  - Attack at $t = 3$, when there are 3 available observations?
  - Or wait until $t = 5$, but invalidating the observations at $t = 4$?



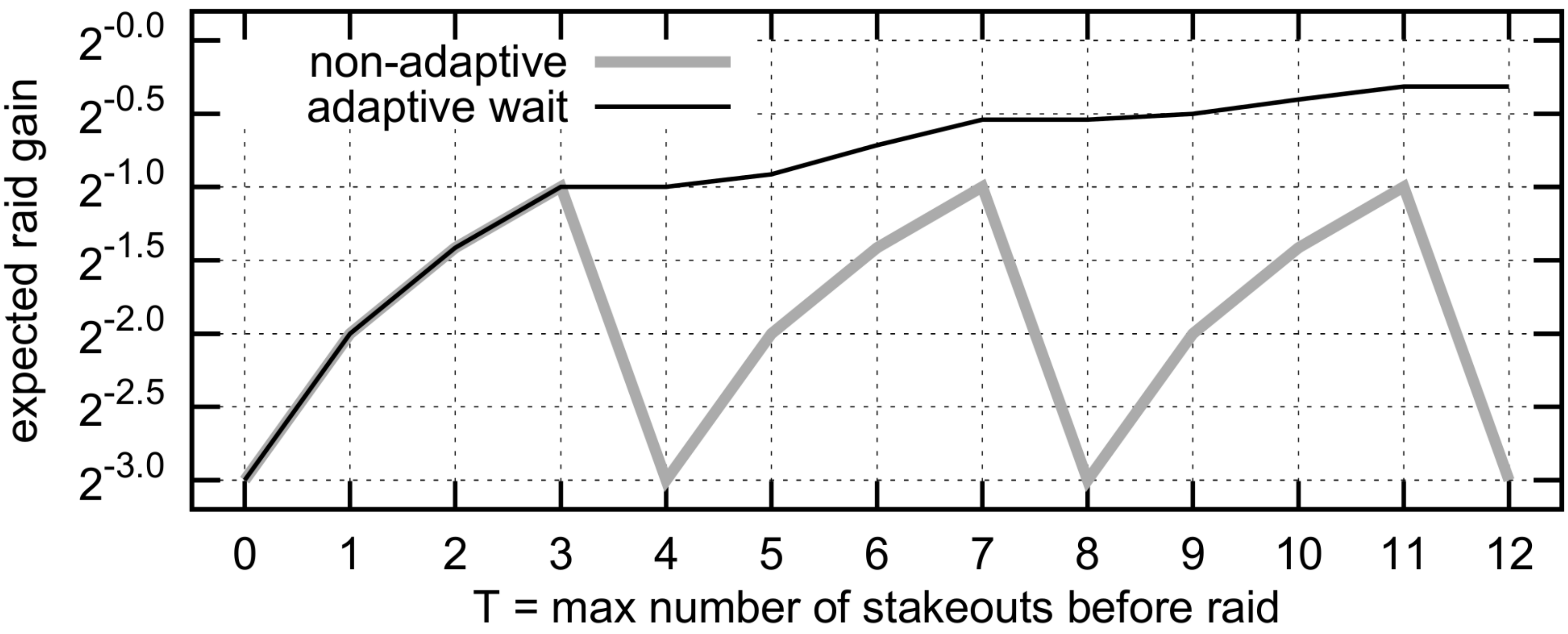
C: Wait-adaptivity

- Attack at $t = 3$:

$$expected\ gain = \frac{1}{5}$$

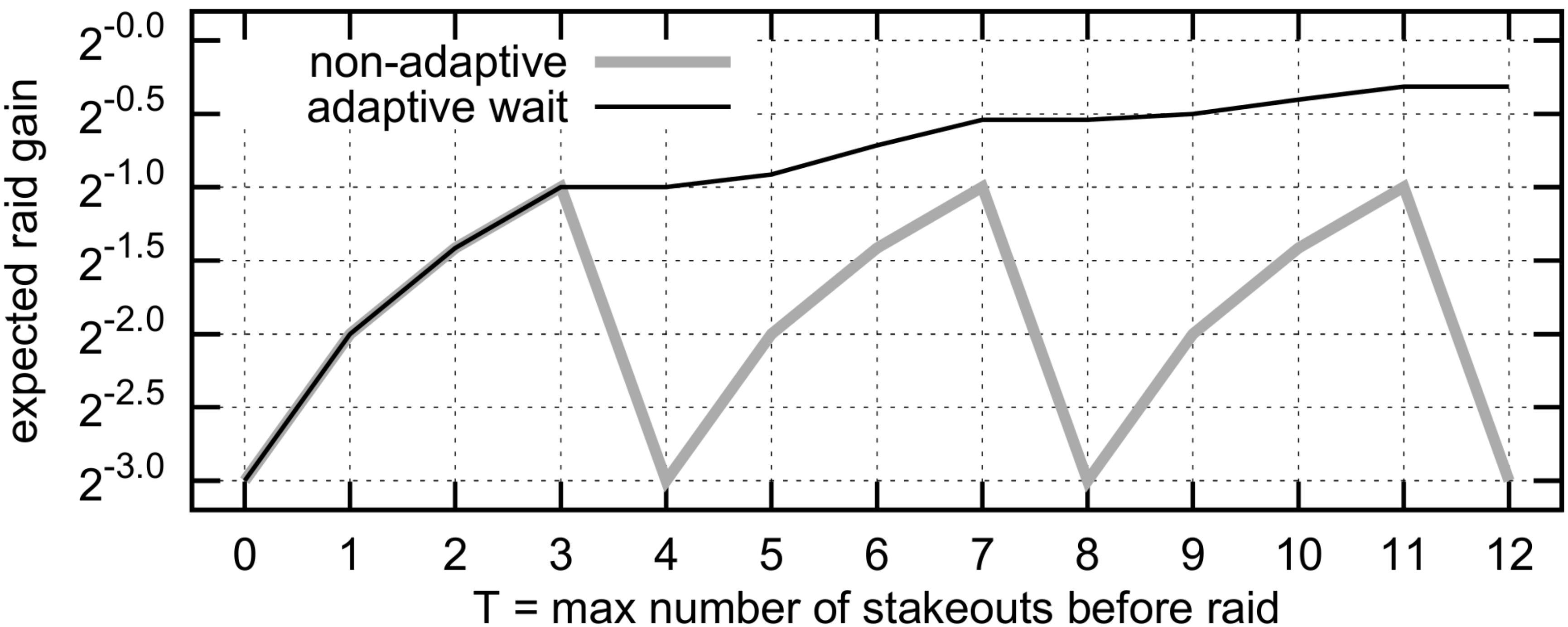
- Attack at $t = 5$:

$$expected\ gain = \frac{1}{8} \cdot 1 + \frac{7}{8} \cdot \frac{1}{7} = \frac{1}{4} > \frac{1}{5}$$



C: Wait-adaptivity

- **Theorem:** Given any gain function that is invariant on the maximum time T, the expected gain D_g at any time t is not greater or equal than the expected gain at time $t+1$.

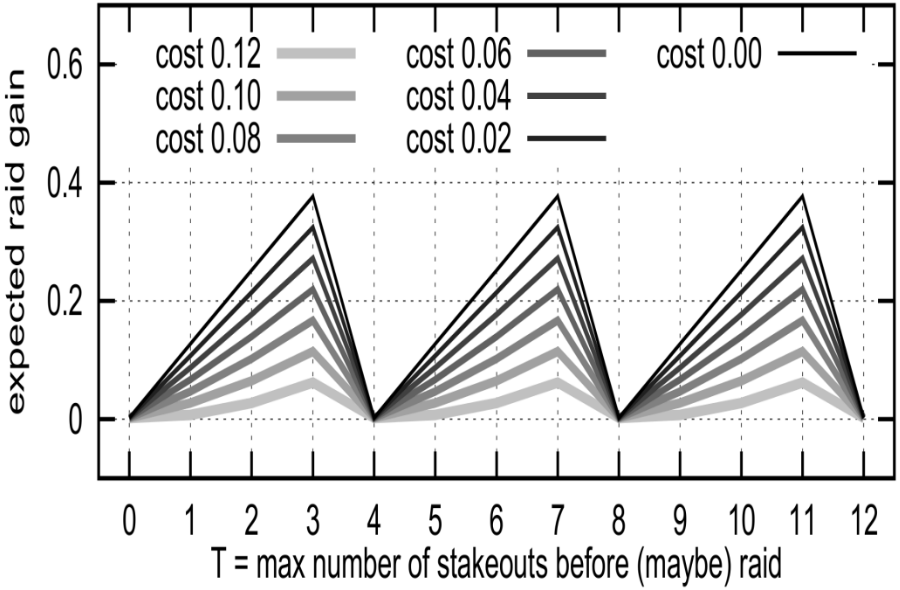


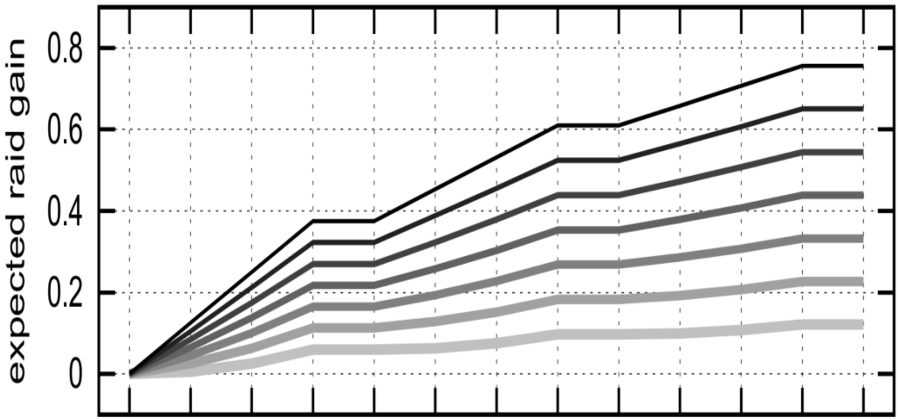
D: Gain can be bounded costly observations

- Each stakeout has cost c applied to final gain.

- Raiding a wrong location is penalized by -1.0

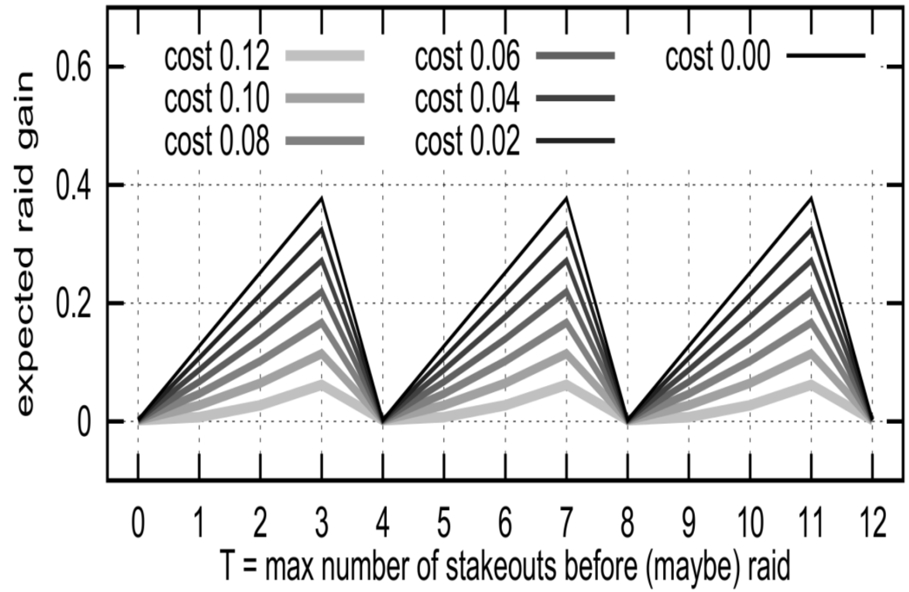
- Not raiding has no penalty.

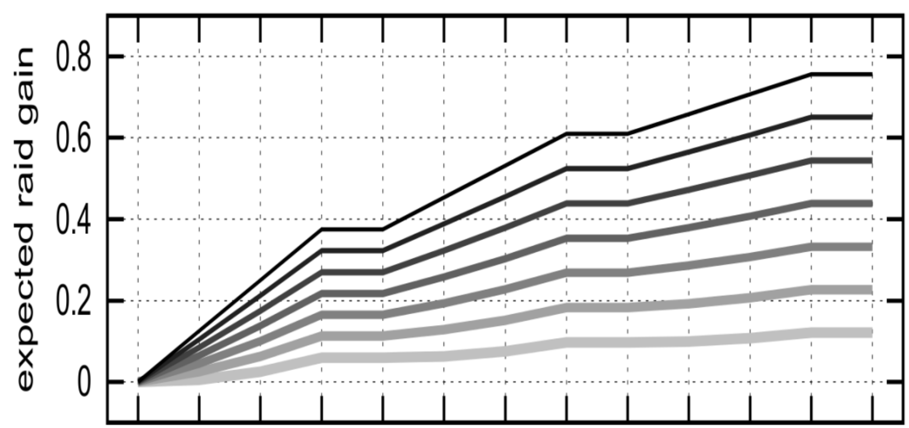




D: Gain can be bounded costly observations

- On top: non wait-adaptive adversary
- On the bottom: wait-adaptive adversary
    - Whenever it is optimal for the adversary to attack at the end of an epoch, it will be so at the end of any epoch.



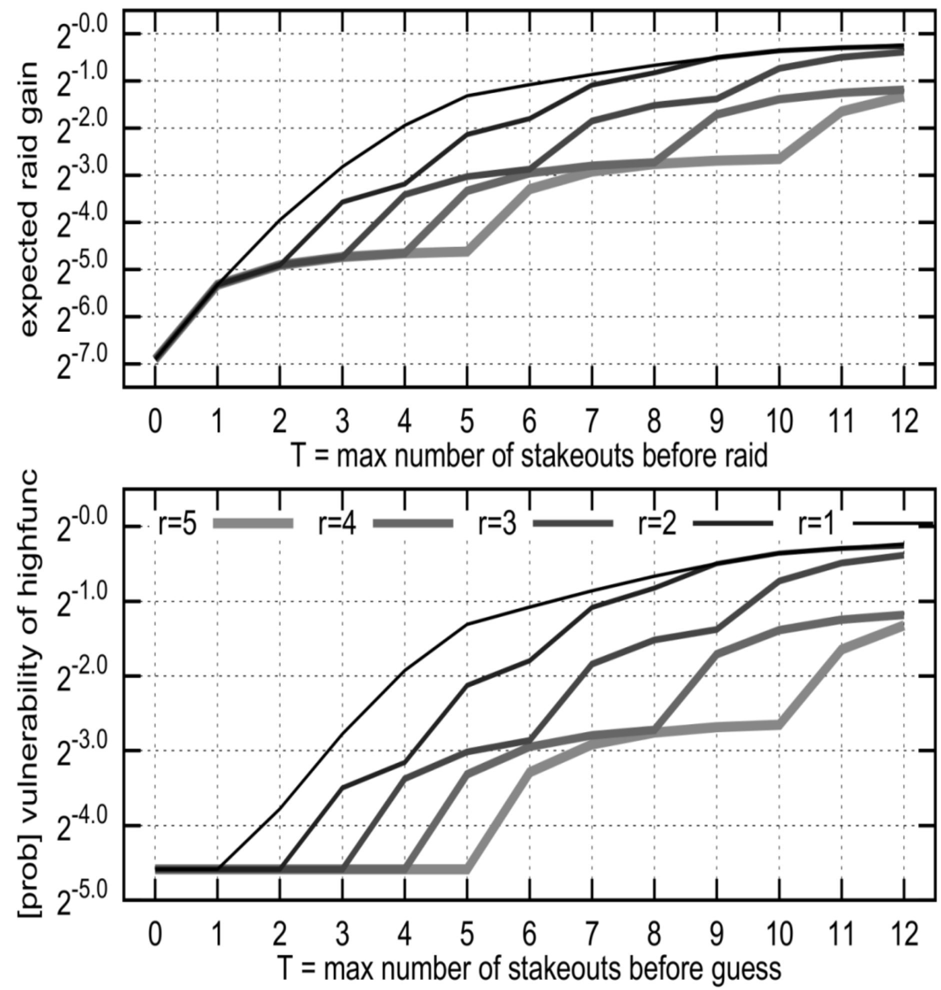


E: Frequent secret change does not necessarily imply less gain

- Consider that there are:
  - n buildings in which the stash can be hidden;
  - $(n-1)!$ floors in each building;
  - each floor is claimed by a drug-dealing gang, and the gang owns the same floor in every building.
- Consider also that the police:
  - Can stake out any building, but is only successful half of the time;
  - Can raid only a particular floor in a particular building (no warranties for a whole building).
- Finally, the each gang moves its stash according to a unique strategy, which is a permutation π of the buildings.

E: Frequent secret change does not necessarily imply less gain

- The chances of successful police raid after a number of stakeouts depend on the change rate r
- Unintuitively, the higher r is, the more leakage.
  - Figure $n = 5\ buildings$
- Our conjecture: the key is the high correlation between secret and secret function.



Current work

- Changing the secret more often is not always preferable to changing it less.
  - We conjecture that such situations require a strong correlation between the secret and the high-input strategy used to evolve the secret.
  - We want to precisely characterize this correlation and the contexts in which it is relevant, so to build more robust systems.

- Our context is more complex than the usual QIF context. We want to understand better how to proceed with a worst-case leakage analysis in our type of context.