Using EasyCrypt to Prove Cryptographic Security

Alley Stoughton
Boston University

Guest Lectures in CS 512, Spring 2018

EasyCrypt

- EasyCrypt is a framework for interactively finding security proofs for cryptographic constructions and protocols
- It is being developed by researchers at IMDEA Software Institute, Inria Sophia-Antipolis and École Polytechnique
  - I'm also playing a role, dating from when I worked for IMDEA; in particular, the reference manual (still a work in progress) is mainly my work
- EasyCrypt has been used to prove the security of a large number of constructions and protocols
- At BU, Ran Canetti, Mayank Varia and I—soon to be joined by Prof. Kfoury—have an ongoing research effort involving EasyCrypt

Help Wanted!

Modeling Cryptographic Security

- Cryptographic constructions and protocols are are probabilistic
- They involve making *random choices* from discrete *(sub-)probability distributions*
  - Each element of a type has a given probability of being chosen—but the sum of all the probabilities may be less than 1
- Cryptographic security is modeled using so-called *games*
- Games are parameterized by *adversaries*, which try to “win” the games

Modeling Cryptographic Security

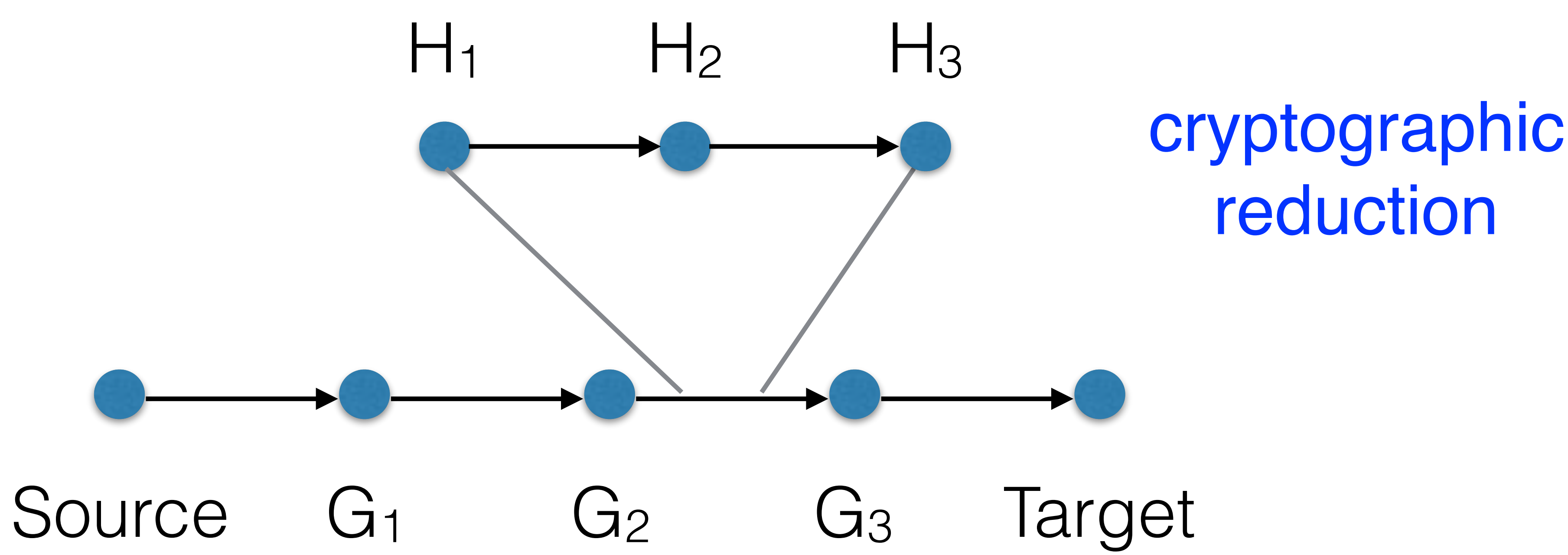
- For example, in IND-CPA (indistinguishability under chosen plaintext attack) security of a symmetric encryption scheme:
    - The adversary chooses two plaintexts, x_1 and x_2
    - The game encrypts one of them, making this choice by flipping a coin, producing the ciphertext c
    - The adversary is given c, and asked to *guess* the result of the game's coin flip
    - The adversary wins, if it guesses correctly—in which case the game returns true, otherwise it returns false
    - The adversary's *advantage* is the absolute value of the difference between the probability of the game returning true and $1/2$
    - A security proof upper-bounds this advantage

Proving Security

- Security can be proved using the *sequence of games* approach
- Source and target games are connected via a series of intermediate games
- In IND-CPA security, the source game described above is connected with a game that can be proved to return true with probability 1/2
- Some game transitions are perfectly secure
- But others incur a penalty—the adversary can distinguish the two games, but by only slightly
- These individual penalties are summed up to give the overall penalty—an upper bound on the absolute value of the difference between IND-CPA game returning true and 1/2

Proving Security

- Game transitions may be proved using *cryptographic reductions*
  - Some reductions are based on *hardness assumptions*
  - Others are proved using their own sequences of games



EasyCrypt's Modules

- In EasyCrypt, cryptographic games are modeled as modules, which consists of global variables and procedures
- Modules may be parameterized, e.g., by adversaries
- Procedures are written in a simple imperative language, with while loops and random assignments

EasyCrypt's Logics

- EasyCrypt has four logics:
  - a **Probabilistic Relational Hoare Logic (pRHL)** for proving relations between pairs of games
  - a **Probabilisitic Hoare logic (pHL)** for proving probabilistic facts about single games
  - an **ordinary Hoare logic (HL)**
  - an **ambient higher-order, classical logic** for proving mathematical facts and connecting judgements from the other logics

EasyCrypt's Proofs and Theories

- Proofs are carried out using tactics, as in Coq
  - A goal consists of hypotheses and a conclusion
  - A tactic reduces a goal to *zero* or more subgoals
- Proofs are developed interactively, but may be replayed step-by-step or checked in batch mode
- Proofs may be structured as sequences of lemmas
- EasyCrypt has an extensive Library
- EasyCrypt theories may be used to group definitions, modules, axioms and lemmas together
- Theories may be specialized via cloning
  - The axioms should then be proved

Plan for Today and Thursday

- In the rest of today's lecture and on Thursday, I'm going to work through several EasyCrypt proofs—interactively using the proof assistant
    - First, we'll prove a fact of elementary logic, showing the equivalence of
        - `forall (x : ‘a), P x`
        - `! exists (x : ‘a), ! P x`
    - Next, we'll prove some facts about two modules making random assignments
        - One returns a random boolean, the other returns the exclusive or of two randomly chosen booleans
    - Finally, we'll define a symmetric encryption scheme based on pseudorandom functions, and prove the IND-CPA security of this scheme

Supporting Resources

- To install EasyCrypt on your computer, follow the instructions at:

  `https://www.easycrypt.info`

- EasyCrypt's reference manual can be found at:

  `https://github.com/EasyCrypt/easycrypt`

- The EasyCrypt proofs we'll be studying can be found at:

  `https://github.com/alleystoughton/EasyTeach`