

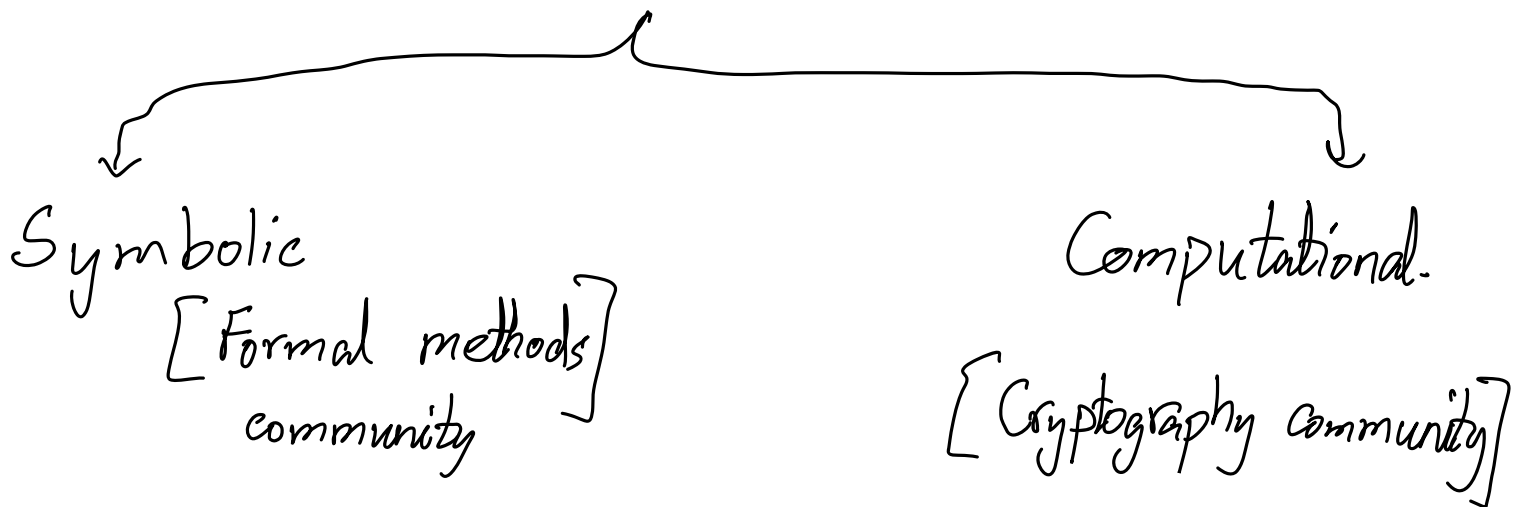
CAC :

- ↳ Design Level
- ↳ Implementation Level.
- ↳ Deployment Level.

Main challenges:

- ↳ Difficult to understand guarantees and fine-print caveats.
- ↳ Broad field, complex and rapidly evolving.

Design level.



Why? Because math is the only proof of security.

Current: Pen and Paper math [Error prone]

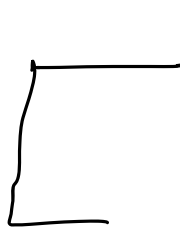
↳ Methods to minimize errors.

↳ Code based game playing ↳ Add realism.
↳ Universal composability
} Still error prone.

Math proofs are hard, hence we need machines that can help. [Halavi 05]

Symbolic Model.

- Primitives are black boxes.
- Terms are atomic
 - ⇒ Adv needs to know full k to decrypt.

Symbolic security  Trace properties.
Equivalence properties

What is the difference? 

Computational Model.

Keys and messages are bitstrings as opposed to blackboxes in symbolic model.

Computational security $\left\{ \begin{array}{l} \rightarrow \text{Game based.} \\ \rightarrow \text{Simulation based.} \end{array} \right.$

Computational security $\left\{ \begin{array}{l} \rightarrow \text{Concrete} \\ \rightarrow \text{Asymptotic.} \end{array} \right.$

Game based. \rightarrow Games between challengers and adversaries.

Methodology \rightarrow Game hopping $\left[\frac{\text{Shoup}}{\text{BR}} \right]$

Simulation based \rightarrow "Real" and "Ideal"

More complicated but support
composition theorems.

Concrete security \rightarrow Involves quantifying the
security by bounding the success prob.
 (t, ϵ) -secure scheme.

Asymptotic security \rightarrow

Views run-time of adversary and
success prob as functions of some parameter

Secure scheme is one which is
broken by polynomial time adv with
negligible prob.

CAC has focused on game-based, concrete.
security notions.