Encryption Cycles and Two Views of Cryptography

Peeter Laud

Tartu Ülikool
Cybernetica AS

$\boxed{\textbf{General problem}}$

- When does a cryptographic protocol achieve its stated objectives?
  - We have in mind *security objectives* — “Certain bad things do not happen”.
- To answer that, we have to fix what a cryptographic protocol is.
- We need
  - A language for expressing the protocols.
  - A meaning (semantics) for that language. That meaning should be
    * realistic, i.e. follow the implementation;
    * easily analysable.

The language

A party of the protocol may

- Receive messages (we will also call them expressions).
- Generate new keys — $K \in \mathbf{Keys}$.
- Construct tuples of messages: $(E_1, E_2, \ldots, E_n)$.
- Encrypt messages with keys: $\{E\}_K$.
- Take components of tuples: E_i from $(E_1, \ldots, E_n)$.
- Decrypt messages: from $\{E\}_K$ and K find E.
- Send messages.

$\boxed{\textbf{The semantics of communication}}$

- The *environment* routes the constructed messages between protocol participants:
  - legitimate parties;
  - the adversary (assumed to have (some) control over the environment).
- The messages come from a certain set.
- Tupling, encryption, etc. are certain operations over that set.
- The powers of the adversary are defined in terms of that set.

Semantics — computational model

- The messages are bit-strings. The set of messages is $\{0,1\}^*$.
- Tupling is some injective, easily computable and invertible function from $(\{0,1\}^*)^*$ to $\{0,1\}^*$.
- Semantics of key generation, encryption and decryption is given by some *encryption scheme* $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ that is supposed to be *secure*.
  - No efficient algorithm can *break* it.
- The adversary can use any efficient algorithm to generate new messages from the ones it has received.
- Such semantics closely models the actual implementation.
- Proofs that some protocol is secure have to be done manually.

Semantics — formal model

- The messages are elements of a formal algebra **Exp**. It contains
  - All elements of the set of keys **Keys**.
  - All elements of some set of *constants* **Consts**.
  - If $E_1, \ldots, E_n \in \mathbf{Exp}$, then **Exp** also contains $(E_1, \ldots, E_n)$.
  - If $E \in \mathbf{Exp}$ and $K \in \mathbf{Keys}$, then **Exp** also contains $\{E\}_K$.
- Semantics of key generation is taking a new element of **Keys**.
- Semantics of tupling and encryption is using the constructors of **Exp**.
- The adversary has the same power as legitimate participants of the protocol.
- There are automatic tools for verifying the correctness of protocols.
- The semantics is quite different from the implementation.

Relationship of two models

- Could we use tools working within the formal model to deduce the security of the protocol in the computational model?
- If the adversary is passive, then the answer is yes.
  (If it is active, then we do not yet know)
- There is one reservation, though — the *encryption cycles.*
- Encryption cycle is a tuple

$$(\{K_2\}_{K_1}, \{K_3\}_{K_2}, \ldots, \{K_n\}_{K_{n-1}}, \{K_1\}_{K_n}) \ .$$

- From this tuple, the keys $K_1, \ldots, K_n$
  - cannot be found in the formal model.
  - possibly can be found in the computational model.
    - Definition of secure encryption says nothing about encryption cycles.

Removing the discrepancy

- We have to mend (at least) one of the models.
- Mending the computational model involves
  - Giving new, stronger definitions of security of encryption schemes.
  - Constructing encryption schemes that satisfy this definition (under some standard assumptions).
- Has been done by Black, Rogaway and Shrimpton; Camenisch and Lysyanskaya.
- For the security of proposed schemes, somewhat more assumptions are necessary than usual.
- Mending the formal model involves
  - Making the adversary more powerful.

$\boxed{\textbf{The capabilities of the adversary — usual definition}}$

The notation $E \vdash E'$ means:

> If the adversary knows the expression E, then it can find the expression E'.

We state the definition of $\vdash$ only for decomposing E.

$$
\begin{aligned}
& E \vdash C \text{ for all } C \in \textbf{Consts} \\
& E \vdash E \\
E \vdash (E_1, E_2) \Longrightarrow\ & E \vdash E_i \\
E \vdash \{E'\}_K \text{ and } E \vdash K \Longrightarrow\ & E \vdash E'
\end{aligned}
$$

$\boxed{\textbf{The capabilities of the strengthened adversary}}$

The notation $E \vdash_{\mathbf{K}} E'$ means (here $\mathbf{K} \subset \mathbf{Keys}$):

> If the adversary knows E and has means to decrypt with all $K \in \mathbf{K}$, then the adversary can find E'.

The relation $\vdash_{\emptyset}$ describes the capabilities of the adversary.

$$
\begin{aligned}
& E \vdash_{\emptyset} C \text{ for all } C \in \mathbf{Consts} \\
& E \vdash_{\emptyset} E \\
E \vdash_{\mathbf{K}} E' \Longrightarrow\ & E \vdash_{\mathbf{K} \cup \mathbf{K}'} E' \\
E \vdash_{\mathbf{K}} (E_1, E_2) \Longrightarrow\ & E \vdash_{\mathbf{K}} E_i \\
E \vdash_{\mathbf{K}} \{E'\}_K \Longrightarrow\ & E \vdash_{\mathbf{K} \cup \{K\}} E' \\
E \vdash_{\mathbf{K} \cup \{K\}} E' \text{ and } E \vdash_{\mathbf{K}} K \Longrightarrow\ & E \vdash_{\mathbf{K}} E' \\
E \vdash_{\mathbf{K} \cup \{K\}} K \Longrightarrow\ & E \vdash_{\mathbf{K}} K
\end{aligned}
$$