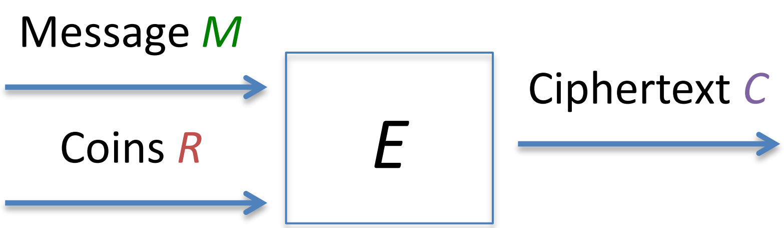
# Standard Security Does Not Imply Security Against Selective-Opening

Mihir Bellare, Rafael Dowsley, Brent Waters, Scott Yilek

(UCSD, UCSD, UT Austin, U. of St. Thomas)

Full version on IACR ePrint archive

# Commitment Schemes




| Sender | | Receiver |
|---|---|---|
| Commit to $M$: | | |
| $R \xleftarrow{\$} \{0,1\}^r$; $C \leftarrow E(M;R)$ | $\xrightarrow{C}$ | |
| Open $M$: | $\xrightarrow{M,\ R}$ | Check that: $C=E(M;R)$ |

Hiding: It is computationally infeasible for the receiver given $C$ to learn anything more about $M$ than it knows a priori. Formalized via semantic security.

Binding: It is computationally infeasible for the sender to find $M$, $M'$, $R$, $R'$ such that: $E(M;R)=E(M';R')$ and $M \neq M'$.

We will call a commitment scheme HB-secure if it satisfies these properties.

# Examples of Commitment Schemes

Commitment Schemes are basic and widely used tools, for example for ZK and other protocols. There are many constructions, for example:

- Let $g$, $h$ be generators of a prime order group $G$ and let $E(M;R)=g^M h^R$ [Ped91].

- Let $H$ be a CR hash function and $Ext$ a strong randomness extractor and let

$$E(M;\, R_1 \| R_2) = (\, H(R_2),\, R_1,\, Ext(R_1, R_2) \oplus M\,).$$

# Hiding

Challenger$_D$ Adversary $A$

$M \leftarrow\$ D$; $R \leftarrow\$ \{0,1\}^r$; $C \leftarrow E(M;R)$ $\xrightarrow{C}$

Security means $A$ cannot figure out anything about $M$. More precisely, anything more than that $M$ is distributed according to $D$. Should hold for all $D$ and is formalized by asking that a simulator denied $C$ does as well as $A$.

# Notation

$\boldsymbol{M}$, $\boldsymbol{R}$, $\boldsymbol{C}$ denote vectors.

$\boldsymbol{M} = (\boldsymbol{M}[1],\ldots, \boldsymbol{M}[n])$ is a vector of messages drawn from a distribution $D$.

$\boldsymbol{R} = (\boldsymbol{R}[1],\ldots, \boldsymbol{R}[n])$ is a vector of independently distributed random strings.

Let $E(\boldsymbol{M};\boldsymbol{R}) = (E(\boldsymbol{M}[1];\boldsymbol{R}[1]),\ldots, E(\boldsymbol{M}[n];\boldsymbol{R}[n]))$.

$\boldsymbol{C} = (\boldsymbol{C}[1],\ldots, \boldsymbol{C}[n]) = E(\boldsymbol{M};\boldsymbol{R})$ is a vector of commitments.

# SOA-M

Challenger$_D$ | Adversary $A$

$\boldsymbol{M} \leftarrow_{\$} D$

$\boldsymbol{R} \leftarrow_{\$} \{0,1\}^{rn}$; $\mathbf{C} \leftarrow E(\boldsymbol{M};\boldsymbol{R})$

$\xrightarrow{\quad \boldsymbol{C} \quad}$

$\xleftarrow{\quad I \subseteq \{1,\ldots,n\} \quad}$

$\xrightarrow{\quad \langle \boldsymbol{M}[i] : i \in I \rangle \quad}$

Security means $A$ cannot figure out anything about $\langle \boldsymbol{M}[i] : i \notin I \rangle$.

Q: If $E$ is HB-secure then is it SOA-M secure?

A: YES.

The proof crucially exploits that $A$ is not given a proof of correct opening.

This version of SOA is called SOA-M and is not the one where difficulties arise.

# SOA-C

Challenger$_D$ Adversary $A$

$\boldsymbol{M} \xleftarrow{\$} D$

$\boldsymbol{R} \xleftarrow{\$} \{0,1\}^{rn}$; $\mathbf{C} \leftarrow E(\boldsymbol{M};\boldsymbol{R})$

$$\xrightarrow{\quad \boldsymbol{C} \quad}$$

$$\xleftarrow{\quad I \subseteq \{1,\ldots,n\} \quad}$$

$$\xrightarrow{\quad \langle \boldsymbol{M}[i], \boldsymbol{R}[i] : i \in I \rangle \quad}$$

Security means $A$ cannot figure out anything about $\langle \boldsymbol{M}[i] : i \notin I \rangle$

Q: If $E$ is HB-secure then is it SOA-C secure?

A: OPEN

- No proof that HB-security implies SOA-C security.
- No counter-example scheme $E$ that is HB-secure but not SOA-C secure.

# Our Results

There exists an HB-secure commitment scheme that is not SOA-C secure.

This answers the long-standing open question. But perhaps the counter-example is artificial. What about “real” schemes?

**Stronger:** Every HB-secure commitment scheme is SOA-C insecure.

In particular the example schemes we gave earlier are not SOA-C secure.

Given any HB-secure scheme we present an attack breaking SOA-C security.

We also show that there exist IND-CPA PKE schemes that are not SOA-C or SOA-K secure, including “real” schemes.

# SOA-C Security

We use the simulation-style definition of SOA-C security introduced by Dwork, Naor, Reingold and Stockmeyer [DNRS03].

Challenger$_{D,Rel}$ | Adversary *A*

$M \xleftarrow{\$} D$

$R \xleftarrow{\$} \{0,1\}^{rn}$

$C \leftarrow E(M;R)$

$\xrightarrow{\quad C \quad}$

$\xleftarrow{\quad I \subseteq \{1,\ldots,n\} \quad}$

$\xrightarrow{\quad \langle M[i], R[i] : i \in I \rangle \quad}$

$\xleftarrow{\quad w \quad}$

*A* wins if $Rel(w, M, I)=1$.

Challenger$_{D,Rel}$ | Simulator *S*

$M \xleftarrow{\$} D$

$\xrightarrow{\quad \wedge \quad}$

$\xleftarrow{\quad I \subseteq \{1,\ldots,n\} \quad}$

$\xrightarrow{\quad \langle M[i] : i \in I \rangle \quad}$

$\xleftarrow{\quad w \quad}$

*S* wins if $Rel(w, M, I)=1$.

$Adv_{E,A,S}$=Pr[*A* wins]- Pr[*S* wins]

*E* is SOA-C secure if for every PT adversary *A* there exists a PT simulator *S* such that $Adv_{E,A,S}$ is negligible in the security parameter.

# Our Result for Commitment

Theorem: Assume CR hash functions exist. Let $E$ be any binding commitment scheme. Then there is a PT message distribution $D$, a PT relation $Rel$ and a PT adversary $A$ such that for every PT simulator $S$ we have:

$$Adv_{E,A,S}(\lambda) \geq 1 - \mathrm{negl}(\lambda)$$

Furthermore the messages output by $D$ are uniformly and independently distributed.

Thus we constructed $D$, $Rel$ such that we can prove there does not exist a efficient simulator, meaning $A$ is a successful attack.

We do not assume simulation is black-box.

# History

SOA-C security for commitment was first defined and considered by Dwork, Naor, Reingold and Stockmeyer [DNRS03].

Hofheinz [Hof11] shows blackbox negative results which indicate it is hard to prove the existence of a SOA-C secure commitment using a blackbox reduction to a standard assumption. This does not say such a scheme doesn’t exist. Potentially a scheme could be proved secure using a non-blackbox reduction or under non-standard assumptions.

Our results are not about the difficulty of proofs or reductions for SOA-C. They are attacks showing secure schemes don’t exist.

# Implications

Our results in particular mean that

$$E(M;R)=g^{M}h^{R}$$

is not SOA-C if the discrete log problem is hard. Same for other commitment schemes.

# Distribution of Messages

Challenger$_{D,Rel}$ | | Adversary $A$
---|---|---
$M \xleftarrow{\$} D$ | |
$R \xleftarrow{\$} \{0,1\}^{rn}$; $C \leftarrow E(M;R)$ | $\xrightarrow{\quad C \quad}$ |
| $\xleftarrow{\quad I \subseteq \{1,\dots,n\} \quad}$ |
| $\xrightarrow{\quad \langle M[i], R[i] : i \in I \rangle \quad}$ |
$A$ wins if $Rel(w, M, I)=1$. | $\xleftarrow{\quad w \quad}$ |

It had been thought that the difficulty in achieving SOA-C was due to the possibility that the messages could be related to each other, but in our attack they are independently and uniformly distributed.

[DNRS03] show that hiding implies SOA-C for independent messages for a restricted version of their main definition where $Rel(w, M, I) = (f(M)=w)$ for some function $f$. Our result implies that this will not extend to their full definition.

No contradiction!

# RO-model

SOA-C secure commitment is achievable in the programmable ROM: $E^H(M;R)=H(R||M)$.

Our results show it is impossible in standard and NPROM models.

Previous separation results:

Nielsen [Nie02] showed that non-committing encryption can be efficiently realized in programmable ROM, but not in standard and NPROM models.

Our separation result is about feasibility, not efficiency.

Dodis, Katz, Smith and Walfish [DKSW09] obtained a (feasibility) separation result for deniable authentication.

# Extensions

The definition of SOA-C we have used is for one-shot adversaries and simulators. Our results extend to the case of adaptive (adversaries and) simulators assuming the messages have super logarithmic length.

# Idea of the Proof

Let $H:\{0,1\}^* \rightarrow \{0,1\}^h$ be a CR hash function. The challenger chooses $n=2h$ uniformly and independently distributed messages.

| $\text{Challenger}_{D,Rel}$ | | Adversary $A$ |
|---|---|---|
| $M \leftarrow\$ D$ | | |
| $R \leftarrow\$ \{0,1\}^{rn}$; $C \leftarrow E(M;R)$ | $\xrightarrow{C}$ | $b[1]\ldots b[h] \leftarrow H(C)$ |
| | $I$ | |

- The adversary corrupts $h$ out of the $n$ senders.
- Whether sender $2i$-1 or $2i$ is corrupted depends on the value of $b[i]$
- The set of corrupted senders is basically an encoding of the hash output.

- Hash constraint: Does the set of corrupted senders correspond to the hash output?
- Opening constraint: Does $C[i]=E(M[i]; R[i])$ for all corrupted senders $i \in I$?

# Idea of the Proof

Note that the specified adversary always makes the relation return true.

Challenger$_{D,Rel}$ $\xrightarrow{\perp}$ Simulator $S$

$\xleftarrow{I \subseteq \{1,\dots,n\}}$

From this state, run two executions of the simulator (sampling different message vectors).

First Execution ($S_1$) | Challenger$_{D,Rel}$ | Second Execution ($S_2$)

$\xleftarrow{\langle \boldsymbol{M}[i] : i \in I \rangle}$ $\boldsymbol{M}, \boldsymbol{M'} \leftarrow_{\$} D$ $\xrightarrow{\langle \boldsymbol{M'}[i] : i \in I \rangle}$

$\xrightarrow{w}$ $\xleftarrow{w'}$

$S_1$ wins if $Rel(w, \boldsymbol{M}, I)$=1. $S_2$ wins if $Rel(w', \boldsymbol{M'}, I)$=1.

The probability of winning the game in both executions and having different message vectors is related to the simulator winning probability through the Reset Lemma [BP02].

If both executions were successful, they must have generated valid $C$, $C'$.

$C \neq C'$ implies a collision in the hash function.

$C = C'$ implies a violation of the binding property.

# SOA-C for Encryption

SOA first arose in the context of encryption [CFGN96]. It was noticed that at the heart of the difficulty was the fact that the encryption functions of most encryption schemes are committing.

For encryption, SOA-C security is achievable by building schemes that are not committing: schemes based on lossy encryption [BHY09, BY09, HLOV11] or deniable encryption [FHKW10, BWY11]. The first solutions were based on non-committing encryption [CFGN96], but these have long keys.

Lead to focus on SOA-C security for commitment, that we showed here it unachievable.

But the basic question remained open:

Is every IND-CPA scheme SOA-C secure? No proof or counter-example.

For example, is ElGamal encryption scheme SOA-C secure?

$E(g^x, M; R) = (g^R, Mg^{xR})$

No proof or attack.

# Our Result for SOA-C Encryption

There exists an encryption scheme that is IND-CPA secure but not SOA-C secure.

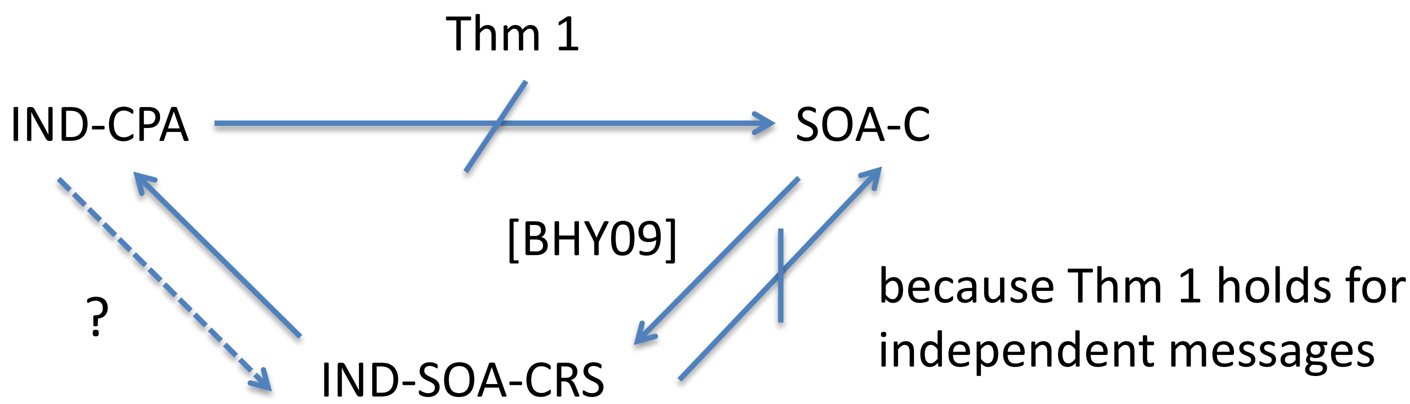
Stronger: Every committing encryption scheme is not SOA-C secure.

We give a precise definition of what being committing means for encryption schemes.

This result includes ElGamal and most standard encryption schemes, so our counter-examples are not artificial.

# Relation to IND

There exists also an indistinguishability-style definition of SOA-security, denoted IND-SOA-CRS, but it is only defined for efficiently conditionally re-samplable distributions.




In a subsequent work, Böhl, Hofheinz and Kraschewski [BHK12] further clarified the relations between the different notions of SOA-security, but the above question remains open.

# Our Result for SOA-K Encryption

Challenger$_D$ Adversary $A$

$M \leftarrow_{\$} D$; $(PK,SK) \leftarrow_{\$} (KG(\lambda))^n$

$R \leftarrow_{\$} \{0,1\}^{rn}$; $C \leftarrow E(PK,M;R)$

$\xrightarrow{C}$

$\xleftarrow{I \subseteq \{1,\dots,n\}}$

$\xrightarrow{\langle M[i], SK[i] : i \in I \rangle}$

We define the notion of decryption verifiability that is a weak form of robustness [ABN10]. For example, ElGamal encryption scheme is not robust, but it is decryption-verifiable.

Our result for SOA-K secure encryption: No decryption-verifiable encryption scheme is SOA-K secure.

# Achievability of SOA-K security

Nielsen [Nie02] showed that any PKE scheme achieving non-committing encryption must have long keys.

$$\begin{tikzcd} \text{NCE} \arrow[r, shift left] & \text{SOA-K} \arrow[l, dotted, shift left, "?"] \end{tikzcd}$$

So conceivably SOA-K can be achieved with short keys. Not ruled out by our above result. But we also show:

Theorem: SOA-K security is impossible with short keys.

Note: this result is not in our abstract/online paper.

# Summary

- Every HB-secure commitment scheme is SOA-C insecure.
- Every committing encryption scheme is SOA-C insecure.
- Schemes specifically designed to achieve SOA-C security are really necessary.
- Every decryption-verifiable encryption scheme is SOA-K insecure.
- SOA-K security needs long keys.

# Thank you!