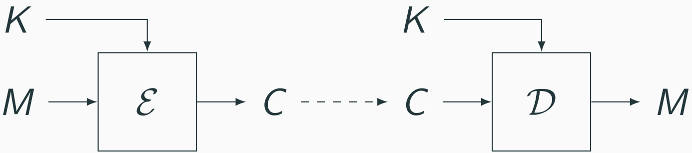
(Im)Possibility of Symmetric Encryption against Coordinated Algorithm Substitution Attacks and Key Exfiltration

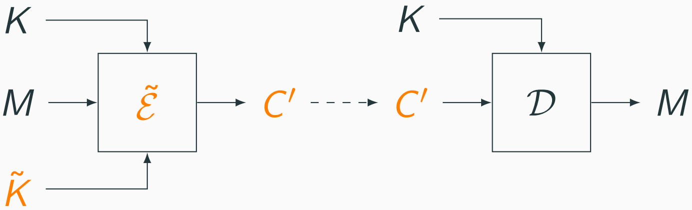
Simone Colombo and Damian Vizàr

LATINCRYPT 2025

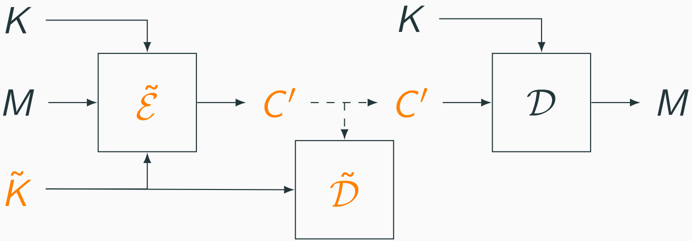
(Im)possibility of simultaneous resistance to ASAs and key exfiltration [BPR14]



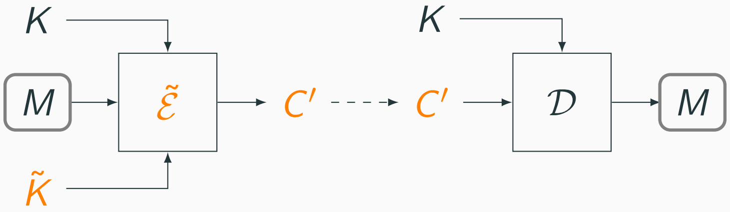
(Im)possibility of simultaneous resistance to ASAs and key exfiltration [BPR14]



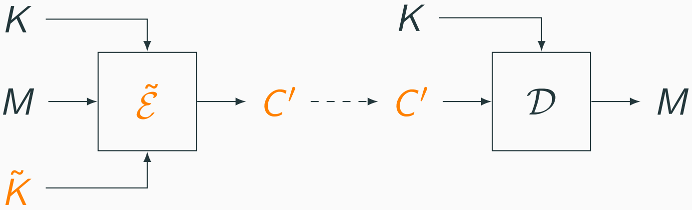
(Im)possibility of simultaneous resistance to ASAs and key exfiltration [BPR14]



(Im)possibility of simultaneous resistance to ASAs and key exfiltration [BPR14]

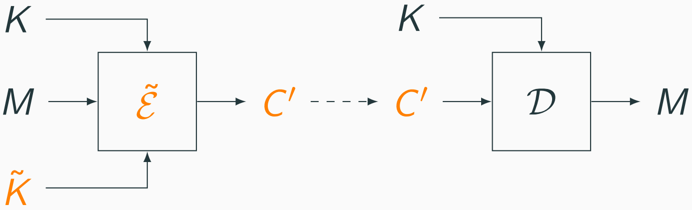


(Im)possibility of simultaneous resistance to ASAs and key exfiltration [BPR14]



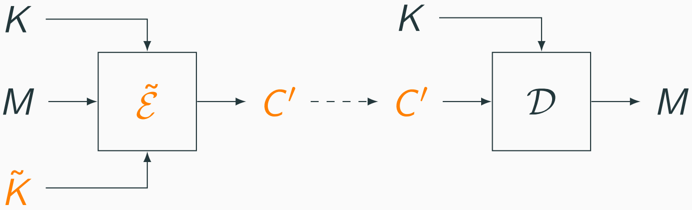
Theorem 4. *Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a unique ciphertext symmetric encryption scheme. Let $\widetilde{\Pi} = (\widetilde{\mathcal{K}}, \widetilde{\mathcal{E}}, \widetilde{\mathcal{D}})$ be a subversion of Π that obeys the decryptability condition relative to Π. Let $\mathscr{B}$ be an adversary. Then* $\mathbf{Adv}^{\mathrm{srv}}_{\Pi,\widetilde{\Pi}}(\mathscr{B}) = 0$.

(Im)possibility of simultaneous resistance to ASAs and key exfiltration [BPR14]



Theorem 4. *Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a unique ciphertext symmetric encryption scheme. Let $\widetilde{\Pi} = (\widetilde{\mathcal{K}}, \widetilde{\mathcal{E}}, \widetilde{\mathcal{D}})$ be a subversion of Π that obeys the decryptability condition relative to Π. Let $\mathscr{B}$ be an adversary. Then* $\mathbf{Adv}^{\mathrm{srv}}_{\Pi,\widetilde{\Pi}}(\mathscr{B}) = 0$.

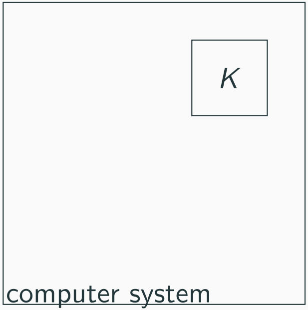
(Im)possibility of simultaneous resistance to ASAs and key exfiltration [BPR14]



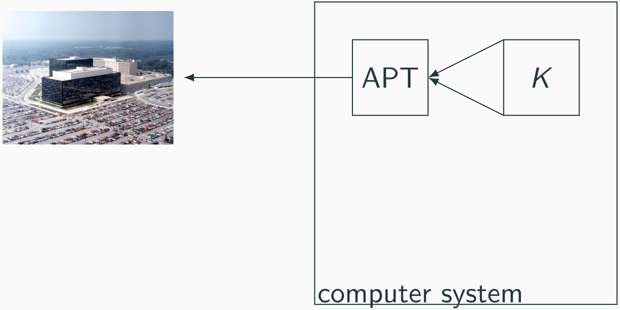
Theorem 4. *Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ be a $\boxed{\text{unique ciphertext symmetric encryption scheme.}}$ Let $\widetilde{\Pi} = (\widetilde{\mathcal{K}}, \widetilde{\mathcal{E}}, \widetilde{\mathcal{D}})$ be a subversion of Π that obeys the decryptability condition relative to Π. Let $\mathscr{B}$ be an adversary. Then* $\mathbf{Adv}^{\mathrm{srv}}_{\Pi,\widetilde{\Pi}}(\mathscr{B}) = 0$.

Due to the correctness condition, any unique-ciphertext scheme is **deterministic**.

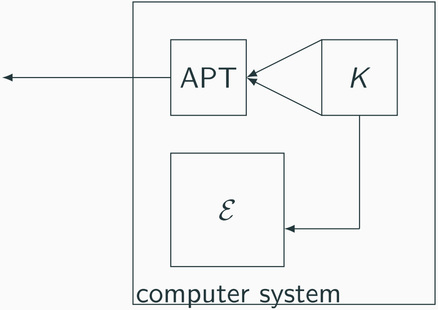
(Im)possibility of simultaneous resistance to ASAs and key exfiltration [BKR16]



(Im)possibility of simultaneous resistance to ASAs and key exfiltration [BKR16]

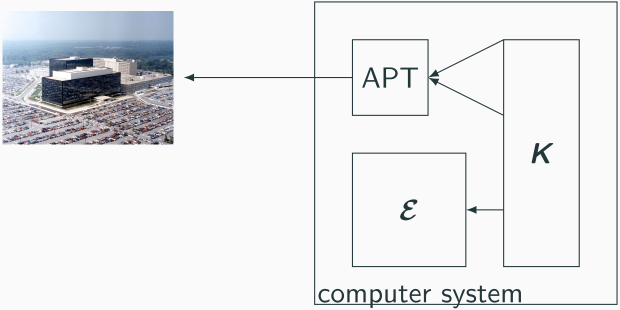


(Im)possibility of simultaneous resistance to ASAs and key exfiltration [BKR16]



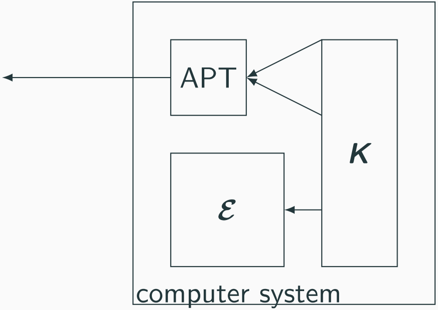
(Im)possibility of simultaneous resistance to ASAs and key exfiltration [BKR16]

$$\mathrm{APT} \approx L \leftarrow_{\$} \mathrm{Lk}^{\mathrm{RO}}(\boldsymbol{K}), \ \text{where } |L| \leq \ell < k, \boldsymbol{K} \in \{0,1\}^k$$



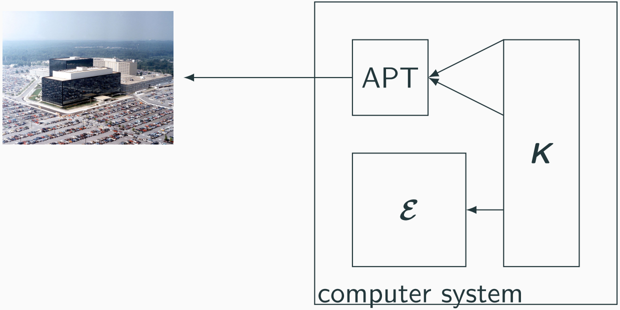
(Im)possibility of simultaneous resistance to ASAs and key exfiltration [BKR16]

$$\mathsf{APT} \approx L \leftarrow\!\!\$\ \mathsf{Lk}^{\mathsf{RO}}(\boldsymbol{K}),\ \text{ where } |L| \leq \ell < k, \boldsymbol{K} \in \{0,1\}^k$$



Algorithm $\mathbf{SE}.\mathsf{Enc}^{\mathrm{RO}}(\boldsymbol{K}, M)$	Algorithm $\mathbf{SE}.\mathsf{Dec}^{\mathrm{RO}}(\boldsymbol{K}, \overline{C})$
$R \twoheadleftarrow \{0,1\}^r$; $K \twoheadleftarrow \mathsf{KEY}^{\mathrm{RO}}(\boldsymbol{K}, R)$ $C \twoheadleftarrow \mathsf{SE.Enc}(K, M)$; $\overline{C} \leftarrow (R, C)$ Return $\overline{C}$	$(R, C) \leftarrow \overline{C}$ $K \leftarrow \mathsf{KEY}^{\mathrm{RO}}(\boldsymbol{K}, R)$ $M \leftarrow \mathsf{SE.Dec}(K, C)$ Return M

$$\mathsf{APT} \approx L \leftarrow\!\!\$\ \mathsf{Lk}^{\mathsf{RO}}(\boldsymbol{K}),\ \text{ where } |L| \leq \ell < k, \boldsymbol{K} \in \{0,1\}^k$$



Algorithm $\mathbf{SE}.\mathsf{Enc}^{\mathrm{RO}}(\boldsymbol{K}, M)$	Algorithm $\mathbf{SE}.\mathsf{Dec}^{\mathrm{RO}}(\boldsymbol{K}, \overline{C})$
$\boxed{R \twoheadleftarrow \{0,1\}^r;}\ K \twoheadleftarrow \mathsf{KEY}^{\mathrm{RO}}(\boldsymbol{K}, R)$	$(R, C) \leftarrow \overline{C}$
$C \twoheadleftarrow \mathsf{SE.Enc}(K, M)\ ;\ \overline{C} \leftarrow (R, C)$	$K \leftarrow \mathsf{KEY}^{\mathrm{RO}}(\boldsymbol{K}, R)$
Return $\overline{C}$	$M \leftarrow \mathsf{SE.Dec}(K, C)$
	Return M

(Im)possibility of simultaneous resistance to ASAs and key exfiltration

Resisting ASAs requires deterministic encryption [BPR14].

Resisting key exfiltration with big keys requires randomized encryption [BKR16].

(Im)possibility of simultaneous resistance to ASAs and key exfiltration

Resisting ASAs requires deterministic encryption [BPR14].

Resisting key exfiltration with big keys requires randomized encryption [BKR16].

"Whether any defense against ASAs is possible in the big-key setting remains open."

[BKR16]

Outline

1. Previous security definitions
2. Security model for simultaneous ASAs and key exfiltration: SURV-LIND
3. Impossibility: generic attack
4. Possibility: big-key encryption with sessions
5. Conclusion and future work

Previous security definitions

Surveillance security for ASAs [BPR14]

Game $\mathrm{SURV}^{\mathscr{B}}_{\Pi,\tilde{\Pi}}$	Oracle $\textsc{Key}(i)$	Oracle $\textsc{Enc}(M, A, i)$
$b \leftarrow\$ \{0,1\}$	**if** $(K_i = \perp)$ **then**	**if** $(K_i = \perp)$ **then return** $\perp$
$\tilde{K} \leftarrow\$ \tilde{\mathcal{K}}$	$\quad K_i \leftarrow\$ \mathcal{K}$	**if** $(b = 1)$ **then** $(C, \sigma_i) \leftarrow\$ \mathcal{E}(K_i, M, A, \sigma_i)$
$b' \leftarrow \mathscr{B}^{\textsc{Key},\textsc{Enc}}(\tilde{K})$	$\quad \sigma_i \leftarrow \varepsilon$	**else** $(C, \sigma_i) \leftarrow\$ \tilde{\mathcal{E}}(\tilde{K}, K_i, M, A, \sigma_i, i)$
return $(b = b')$	**return** ε	**return** C

Adversary $\mathscr{B}$ with master key $\tilde{K}$ must distinguish between correct $\mathcal{E}$ and subverted $\tilde{\mathcal{E}}$.

Another SURV definition [this work]

Game $\mathrm{SURV\$}^{\mathscr{B}}_{\Pi,\tilde{\Pi}}$	Oracle $\textsc{Key}(i)$	Oracle $\textsc{Enc}(M, A, i)$
$b \leftarrow\$ \{0,1\}$	**if** $(K_i = \perp)$ **then**	**if** $(K_i = \perp)$ **then return** $\perp$
$\tilde{K} \leftarrow\$ \tilde{\mathcal{K}}$	$\quad K_i \leftarrow\$ \mathcal{K}$	$(C_0, \sigma_i) \leftarrow\$ \tilde{\mathcal{E}}(\tilde{K}, K_i, M, A, \sigma_i, i)$
$b' \leftarrow \mathscr{B}^{\textsc{Key},\textsc{Enc}}(\tilde{K})$	$\quad \sigma_i \leftarrow \varepsilon$	$C_1 \leftarrow\$ \{0,1\}^{\lvert C_0 \rvert}$
return $(b = b')$	**return** ε	**return** C_b

$\mathscr{B}$ with $\tilde{K}$ must distinguish between random C_1 and C_0 that the subverted $\tilde{\mathcal{E}}$ returns.

Another SURV definition [this work]

Game $\mathrm{SURV\$}^{\mathscr{B}}_{\Pi,\tilde{\Pi}}$	Oracle $\textsc{Key}(i)$	Oracle $\textsc{Enc}(M, A, i)$
$b \leftarrow\$ \{0,1\}$	**if** $(K_i = \perp)$ **then**	**if** $(K_i = \perp)$ **then return** $\perp$
$\tilde{K} \leftarrow\$ \tilde{\mathcal{K}}$	$\quad K_i \leftarrow\$ \mathcal{K}$	$(C_0, \sigma_i) \leftarrow\$ \tilde{\mathcal{E}}(\tilde{K}, K_i, M, A, \sigma_i, i)$
$b' \leftarrow \mathscr{B}^{\textsc{Key},\textsc{Enc}}(\tilde{K})$	$\quad \sigma_i \leftarrow \varepsilon$	$C_1 \leftarrow\$ \{0,1\}^{\lvert C_0 \rvert}$
return $(b = b')$	**return** ε	**return** C_b

$\mathscr{B}$ with $\tilde{K}$ must distinguish between random C_1 and C_0 that the subverted $\tilde{\mathcal{E}}$ returns.

We prove that SURV\$ $\xLeftrightarrow{\text{IND\$-CPA}}$ SURV.

Indistinguishability in presence of leakage for key exfiltration [BKR16]

Game $\mathrm{LIND}_{\mathbf{\Pi}}^{\mathscr{B}}$	Oracle $\textsc{Enc}(M_0, M_1)$
$(\mathsf{Lk}, \sigma) \leftarrow \mathscr{B}^{\mathsf{RO}}$	$C \leftarrow\$ \boldsymbol{\mathcal{E}}^{\mathsf{RO}}(\boldsymbol{K}, M_b)$
$\boldsymbol{K} \leftarrow\$ \{0,1\}^k$	**return** C
$L \leftarrow \mathsf{Lk}^{\mathsf{RO}}(\boldsymbol{K})$	
$b \leftarrow\$ \{0,1\}$	
$b' \leftarrow \mathscr{B}^{\textsc{Enc},\mathsf{RO}}(L, \sigma)$	
return $(b = b')$	

Classic left-or-right IND-CPA game, taking leakage $L \leftarrow \mathsf{Lk}^{\mathsf{RO}}(\boldsymbol{K})$ into account.

Another LIND definition [this work]

Game $\mathrm{LIND\$}_{\mathbf{\Pi}}^{\mathscr{B}}$	Oracle $\textsc{Enc}(M)$
$(\mathrm{Lk}, \sigma) \leftarrow \mathscr{B}^{\mathrm{RO}}$	$C_0 \leftarrow\!\!\$\ \boldsymbol{\mathcal{E}}^{\mathrm{RO}}(\boldsymbol{K}, M)$
$\boldsymbol{K} \leftarrow\!\!\$\ \{0,1\}^k$	$C_1 \leftarrow\!\!\$\ \{0,1\}^{\lvert C_0 \rvert}$
$L \leftarrow \mathrm{Lk}^{\mathrm{RO}}(\boldsymbol{K})$	**return** C_b
$b \leftarrow\!\!\$\ \{0,1\}$	
$b' \leftarrow \mathscr{B}^{\textsc{Enc},\mathrm{RO}}(L, \sigma)$	
return $(b = b')$	

Classic IND\$-CPA game, taking leakage $L \leftarrow \mathrm{Lk}^{\mathrm{RO}}(\boldsymbol{K})$ into account.

Another LIND definition [this work]

Game $\text{LIND\$}_{\Pi}^{\mathscr{B}}$	Oracle $\textsc{Enc}(M)$
$(\mathsf{Lk}, \sigma) \leftarrow \mathscr{B}^{\mathsf{RO}}$	$C_0 \leftarrow_{\$} \mathcal{E}^{\mathsf{RO}}(\boldsymbol{K}, M)$
$\boldsymbol{K} \leftarrow_{\$} \{0,1\}^k$	$C_1 \leftarrow_{\$} \{0,1\}^{\lvert C_0 \rvert}$
$L \leftarrow \mathsf{Lk}^{\mathsf{RO}}(\boldsymbol{K})$	**return** C_b
$b \leftarrow_{\$} \{0,1\}$	
$b' \leftarrow \mathscr{B}^{\textsc{Enc},\mathsf{RO}}(L, \sigma)$	
return $(b = b')$	

Classic IND\$-CPA game, taking leakage $L \leftarrow \mathsf{Lk}^{\mathsf{RO}}(\boldsymbol{K})$ into account.

We show that LIND\$ $\implies$ LIND.

Security model for simultaneous ASAs and KE

Security model for simultaneous ASAs and key exfiltration [this work]

Game $\text{SURV-LIND}^{\mathscr{B}}_{\mathbf{\Pi},\tilde{\mathbf{\Pi}}}$	Oracle $\textsc{Enc}(M, A, i)$	Oracle $\textsc{Leak}(i)$
$\tilde{K} \leftarrow\$ \tilde{\mathcal{K}}$	**if** $(\boldsymbol{K}_i = \perp)$ **then return** $\perp$	**if** $(\boldsymbol{K}_i = \perp)$ **then**
$(\mathsf{Lk}, \sigma) \leftarrow \mathscr{B}^{\mathsf{RO}}(\tilde{K})$	$(C_0, \sigma_i) \leftarrow\$ \tilde{\boldsymbol{\mathcal{E}}}^{\mathsf{RO}}(\tilde{K}, \boldsymbol{K}_i, M, A, \sigma_i, i)$	$\quad \boldsymbol{K}_i \leftarrow\$ \{0,1\}^k$
$b \leftarrow\$ \{0,1\}$	$C_1 \leftarrow\$ \{0,1\}^{\lvert C_0 \rvert}$	$\quad \sigma_i \leftarrow \varepsilon$
$b' \leftarrow \mathscr{B}^{\textsc{Leak},\textsc{Enc},\mathsf{RO}}(\tilde{K}, \sigma)$	**return** C_b	$\quad L \leftarrow\$ \mathsf{Lk}^{\mathsf{RO}}(\boldsymbol{K}_i)$
return $(b = b')$		$\quad$ **return** L
		return $\perp$

Security model for simultaneous ASAs and key exfiltration [this work]

Game $\text{SURV-LIND}^{\mathscr{B}}_{\mathbf{\Pi},\tilde{\mathbf{\Pi}}}$	Oracle $\textsc{Enc}(M, A, i)$	Oracle $\textsc{Leak}(i)$
$\tilde{K} \leftarrow\$ \tilde{\mathcal{K}}$	**if** $(\boldsymbol{K}_i = \perp)$ **then return** $\perp$	**if** $(\boldsymbol{K}_i = \perp)$ **then**
$(\mathsf{Lk}, \sigma) \leftarrow \mathscr{B}^{\mathsf{RO}}(\tilde{K})$	$(C_0, \sigma_i) \leftarrow\$ \tilde{\mathcal{E}}^{\mathsf{RO}}(\tilde{K}, \boldsymbol{K}_i, M, A, \sigma_i, i)$	$\quad \boldsymbol{K}_i \leftarrow\$ \{0,1\}^k$
$b \leftarrow\$ \{0,1\}$	$C_1 \leftarrow\$ \{0,1\}^{\lvert C_0 \rvert}$	$\quad \sigma_i \leftarrow \varepsilon$
$b' \leftarrow \mathscr{B}^{\textsc{Leak},\textsc{Enc},\mathsf{RO}}(\tilde{K}, \sigma)$	**return** C_b	$\quad L \leftarrow\$ \mathsf{Lk}^{\mathsf{RO}}(\boldsymbol{K}_i)$
return $(b = b')$		$\quad$ **return** L
		return $\perp$

As in SURV\$ (equiv. to SURV) distinguish between random C_1 and C_0 from $\tilde{\mathcal{E}}$, with leakage $L \leftarrow\$ \mathsf{Lk}^{\mathsf{RO}}(\boldsymbol{K})$ as in LIND\$ (equiv. to LIND) through $\textsc{Leak}$ oracle.

Security model for simultaneous ASAs and key exfiltration [this work]

Game $\text{SURV-LIND}^{\mathscr{B}}_{\mathbf{\Pi},\tilde{\mathbf{\Pi}}}$	Oracle $\textsc{Enc}(M, A, i)$	Oracle $\textsc{Leak}(i)$
$\tilde{K} \leftarrow\!\!\$\ \tilde{\mathcal{K}}$	**if** $(\boldsymbol{K}_i = \perp)$ **then return** $\perp$	**if** $(\boldsymbol{K}_i = \perp)$ **then**
$(\mathsf{Lk}, \sigma) \leftarrow \mathscr{B}^{\mathsf{RO}}(\tilde{K})$	$(C_0, \sigma_i) \leftarrow\!\!\$\ \tilde{\boldsymbol{\mathcal{E}}}^{\mathsf{RO}}(\tilde{K}, \boldsymbol{K}_i, M, A, \sigma_i, i)$	$\quad \boldsymbol{K}_i \leftarrow\!\!\$\ \{0,1\}^k$
$b \leftarrow\!\!\$\ \{0,1\}$	$C_1 \leftarrow\!\!\$\ \{0,1\}^{\lvert C_0 \rvert}$	$\quad \sigma_i \leftarrow \varepsilon$
$b' \leftarrow \mathscr{B}^{\textsc{Leak},\textsc{Enc},\mathsf{RO}}(\tilde{K}, \sigma)$	**return** C_b	$\quad L \leftarrow\!\!\$\ \mathsf{Lk}^{\mathsf{RO}}(\boldsymbol{K}_i)$
return $(b = b')$		$\quad$ **return** L
		return $\perp$

As in SURV$ (equiv. to SURV) distinguish between random C_1 and C_0 from $\tilde{\boldsymbol{\mathcal{E}}}$,
with leakage $L \leftarrow\!\!\$\ \mathsf{Lk}^{\mathsf{RO}}(\boldsymbol{K})$ as in LIND$ (equiv. to LIND) through Leak oracle.

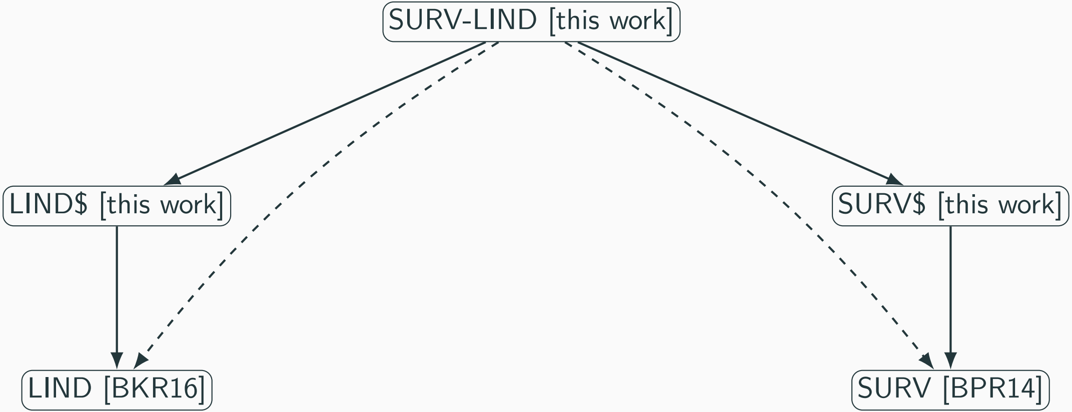
Security model for simultaneous ASAs and key exfiltration [this work]

Game $\text{SURV-LIND}^{\mathscr{B}}_{\mathbf{\Pi},\tilde{\mathbf{\Pi}}}$	Oracle $\textsc{Enc}(M, A, i)$	Oracle $\textsc{Leak}(i)$
$\tilde{K} \leftarrow\$ \tilde{\mathcal{K}}$	**if** $(\boldsymbol{K}_i = \bot)$ **then return** $\bot$	**if** $(\boldsymbol{K}_i = \bot)$ **then**
$(\mathsf{Lk}, \sigma) \leftarrow \mathscr{B}^{\mathsf{RO}}(\tilde{K})$	$(C_0, \sigma_i) \leftarrow\$ \tilde{\boldsymbol{\mathcal{E}}}^{\mathsf{RO}}(\tilde{K}, \boldsymbol{K}_i, M, A, \sigma_i, i)$	$\quad \boldsymbol{K}_i \leftarrow\$ \{0,1\}^k$
$b \leftarrow\$ \{0,1\}$	$C_1 \leftarrow\$ \{0,1\}^{\lvert C_0 \rvert}$	$\quad \sigma_i \leftarrow \varepsilon$
$b' \leftarrow \mathscr{B}^{\textsc{Leak},\textsc{Enc},\mathsf{RO}}(\tilde{K}, \sigma)$	**return** C_b	$\quad L \leftarrow\$ \mathsf{Lk}^{\mathsf{RO}}(\boldsymbol{K}_i)$
return $(b = b')$		$\quad$ **return** L
		return $\bot$

As in SURV\$ (equiv. to SURV) distinguish between random C_1 and C_0 from $\tilde{\boldsymbol{\mathcal{E}}}$, with leakage $L \leftarrow\$ \mathsf{Lk}^{\mathsf{RO}}(\boldsymbol{K})$ as in LIND\$ (equiv. to LIND) through $\textsc{Leak}$ oracle.

We show that SURV-LIND $\Rightarrow$ LIND\$ and SURV-LIND $\Rightarrow$ SURV\$.

Summary of security notions



Solid: proved.   Dashed: by transitivity.

Impossibility: generic attack

Generic attack: leakage function

Algorithm $\mathsf{Lk}^{\mathsf{RO}}_{i,\tilde{K},M}(\boldsymbol{K}_i)$
// state management
$r \leftarrow \mathsf{RO}(\langle i, \tilde{K}, 0\rangle, \lvert r\rvert)$
$(C, \sigma') \leftarrow \boldsymbol{\mathcal{E}}^{\mathsf{RO}}(\boldsymbol{K}_i, M, \varepsilon, \sigma; r)$
return $\mathsf{RO}(\langle C\rangle, \ell)$

Returns the ℓ-bits “hash” of the ciphertext from the encryption of M with coins r.

Generic attack: subversion

Algorithm $\tilde{\boldsymbol{\mathcal{E}}}^{\mathrm{RO}}(\tilde{K}, \boldsymbol{K}_i, M, A, \sigma, i)$

// state management where σ parses as $\tilde{\sigma}, \bar{\sigma}$

$r \leftarrow \mathrm{RO}(\langle i, \tilde{K}, \tilde{\sigma}\rangle, |r|)$

$(C, \bar{\sigma}) \leftarrow \boldsymbol{\mathcal{E}}^{\mathrm{RO}}(\boldsymbol{K}_i, M, A, \sigma; r)$

return $C, \langle\tilde{\sigma}, \bar{\sigma}\rangle$

Returns the ciphertext of M under the same coins r used by the leakage function.

Generic attack

Algorithm $\mathscr{B}_{\mathsf{drnd}}(\tilde{K}, \tau)$	Algorithm $\tilde{\boldsymbol{\mathcal{E}}}^{\mathsf{RO}}(\tilde{K}, \boldsymbol{K}_i, M, A, \sigma, i)$
if $(\tau = \bot)$ **then**	// state management where σ parses as $\tilde{\sigma}, \bar{\sigma}$
$\quad M \leftarrow\$ \{0,1\}^{\nu}$	$r \leftarrow \mathsf{RO}(\langle i, \tilde{K}, \tilde{\sigma}\rangle, \lvert r \rvert)$
$\quad$ **return** $(\mathsf{Lk}_{i,\tilde{K},M}, M)$	$(C, \bar{\sigma}) \leftarrow \boldsymbol{\mathcal{E}}^{\mathsf{RO}}(\boldsymbol{K}_i, M, A, \sigma; r)$
else	**return** $C, \langle \tilde{\sigma}, \bar{\sigma}\rangle$
$\quad M \leftarrow \tau$	
$i \leftarrow\$ \mathcal{I};\ A \leftarrow \varepsilon$	Algorithm $\mathsf{Lk}^{\mathsf{RO}}_{i,\tilde{K},M}(\boldsymbol{K}_i)$
$L \leftarrow \textsc{Leak}(i)$	// state management
$C \leftarrow \textsc{Enc}(M, A, i)$	$r \leftarrow \mathsf{RO}(\langle i, \tilde{K}, 0\rangle, \lvert r \rvert)$
$b' \leftarrow (L \neq \mathsf{RO}(\langle C\rangle, \ell))$	$(C, \sigma') \leftarrow \boldsymbol{\mathcal{E}}^{\mathsf{RO}}(\boldsymbol{K}_i, M, \varepsilon, \sigma; r)$
return b'	**return** $\mathsf{RO}(\langle C\rangle, \ell)$

Adversary $\mathscr{B}$ gets ℓ-bit "hash" L, queries $\textsc{Enc}$ to get C and checks $(L \neq \mathsf{RO}(\langle C\rangle, \ell))$.

Generic attack: informal summary

The subversion's control of random coins lets the leakage precompute the ciphertext.

Possibility: big-key encryption with sessions

Motivation

Problem: ASA $\implies$ force usage of predefined coins

KE $\implies$ leakage can precompute ciphertext.

$\implies$ complete control.

Motivation

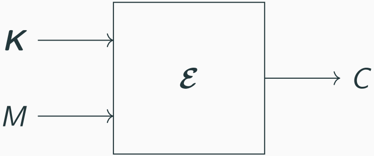
Problem: ASA $\implies$ force usage of predefined coins

KE $\implies$ leakage can precompute ciphertext.

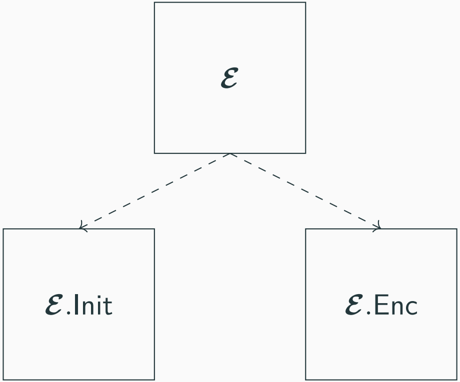
$\implies$ complete control.

Solution: Secure randomness generation.

Big-key symmetric encryption with sessions



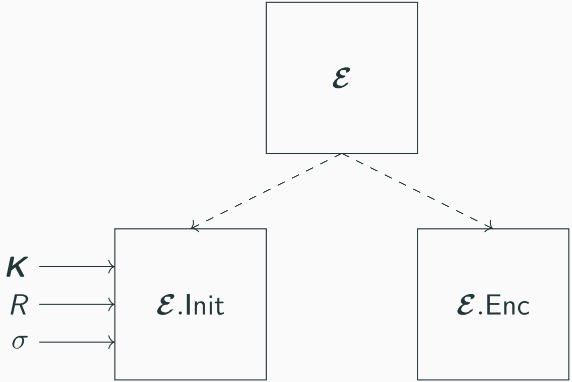
Big-key symmetric encryption with sessions



big-key encapsulation

encryption

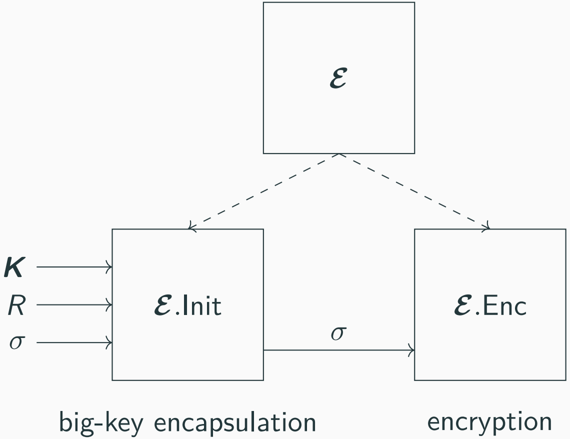
Big-key symmetric encryption with sessions



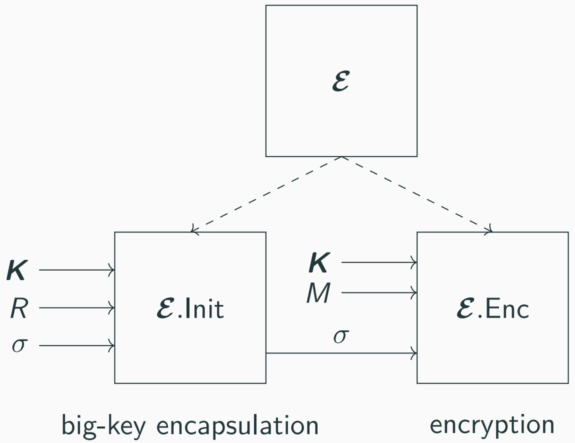
big-key encapsulation

encryption

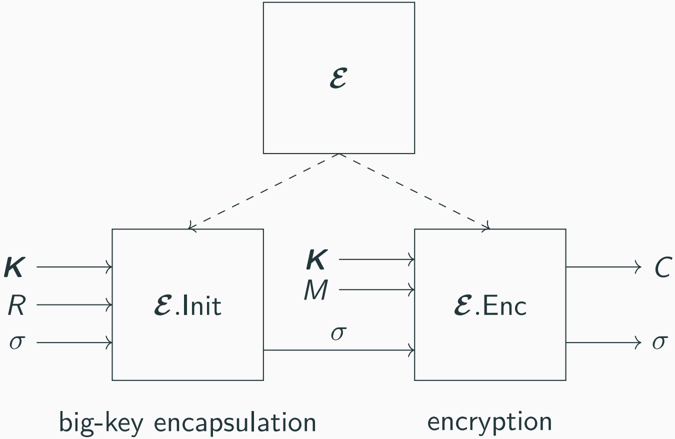
Big-key symmetric encryption with sessions



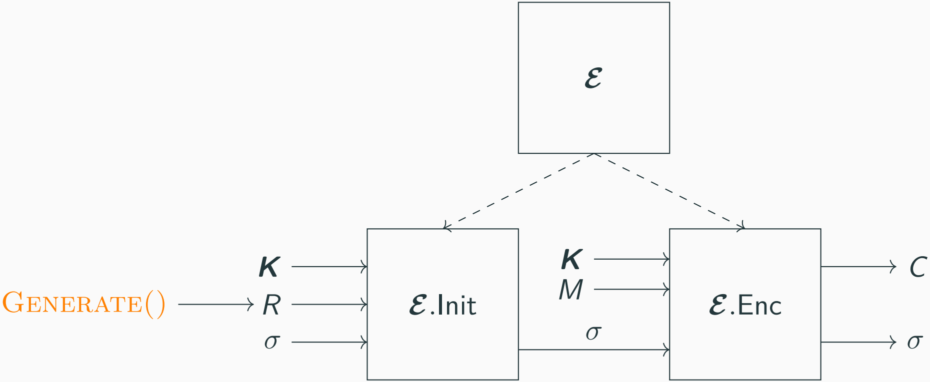
Big-key symmetric encryption with sessions



Big-key symmetric encryption with sessions



Big-key symmetric encryption with sessions



big-key encapsulation

encryption

needs secure randomness

Security notion: RESIST

Game $\text{RESIST}^{\mathscr{B}}_{\mathbf{\Pi},\tilde{\mathbf{\Pi}}}$

$\overrightarrow{\boldsymbol{K}}_i, \sigma_i \leftarrow \perp$ everywhere

$\tilde{K} \leftarrow\$ \tilde{\mathcal{K}}$

$(\mathsf{Lk}, \tau) \leftarrow\$ \mathscr{B}^{\mathsf{RO}}(\tilde{K})$

$b \leftarrow\$ \{0,1\}$

$b' \leftarrow\$ \mathscr{B}^{\text{LEAK},\text{INIT},\text{ENC},\mathsf{RO}}(\tilde{K}, \tau)$

return $(b = b')$

Oracle $\text{LEAK}(i)$

if $(\boldsymbol{K}_i \neq \perp)$ **then return** $\perp$

$\boldsymbol{K}_i \leftarrow\$ \boldsymbol{\mathcal{K}}$; $L \leftarrow\$ \mathsf{Lk}^{\mathsf{RO}}(\boldsymbol{K}_i)$

return L

Oracle $\text{ENC}(M, A, i)$

if $(\sigma_i = \perp)$ **then return** $\perp$

$C_0, \sigma_i \leftarrow\$ \tilde{\boldsymbol{\mathcal{E}}}.\mathsf{Enc}^{\mathsf{RO}}(\tilde{K}, \boldsymbol{K}_i, M, A, \sigma_i, i)$

$C_1 \leftarrow\$ \{0,1\}^{|C_0|}$

return C_b

Oracle $\text{INIT}(i)$

if $(\boldsymbol{K}_i = \perp)$ **then return** $\perp$

$R \leftarrow\$ \text{GENERATE}()$

if $R = \perp$ **then** abort

$\sigma_i \leftarrow\$ \tilde{\boldsymbol{\mathcal{E}}}.\mathsf{Init}^{\mathsf{RO}}(\tilde{K}, \boldsymbol{K}_i, R, \sigma_i, i)$

return R

Proc. $\text{GENERATE}()$

$R \leftarrow\$ \mathcal{R}$

return R

Security notion: RESIST

Game $\mathrm{RESIST}^{\mathscr{B}}_{\mathbf{\Pi},\tilde{\mathbf{\Pi}}}$

$\overrightarrow{\boldsymbol{K}}_i, \sigma_i \leftarrow \perp$ everywhere

$\tilde{K} \leftarrow\!\!\$\ \tilde{\mathcal{K}}$

$(\mathsf{Lk}, \tau) \leftarrow\!\!\$\ \mathscr{B}^{\mathsf{RO}}(\tilde{K})$

$b \leftarrow\!\!\$\ \{0,1\}$

$b' \leftarrow\!\!\$\ \mathscr{B}^{\textsc{Leak},\textsc{Init},\textsc{Enc},\mathsf{RO}}(\tilde{K}, \tau)$

return $(b = b')$

Oracle $\textsc{Leak}(i)$

if $(\boldsymbol{K}_i \neq \perp)$ **then return** $\perp$

$\boldsymbol{K}_i \leftarrow\!\!\$\ \boldsymbol{\mathcal{K}}$; $L \leftarrow\!\!\$\ \mathsf{Lk}^{\mathsf{RO}}(\boldsymbol{K}_i)$

return L

Oracle $\textsc{Enc}(M, A, i)$

if $(\sigma_i = \perp)$ **then return** $\perp$

$C_0, \sigma_i \leftarrow\!\!\$\ \tilde{\boldsymbol{\mathcal{E}}}.\mathsf{Enc}^{\mathsf{RO}}(\tilde{K}, \boldsymbol{K}_i, M, A, \sigma_i, i)$

$C_1 \leftarrow\!\!\$\ \{0,1\}^{|C_0|}$

return C_b

Oracle $\textsc{Init}(i)$

if $(\boldsymbol{K}_i = \perp)$ **then return** $\perp$

$R \leftarrow\!\!\$\ \textsc{Generate}()$

if $R = \perp$ **then** abort

$\sigma_i \leftarrow\!\!\$\ \tilde{\boldsymbol{\mathcal{E}}}.\mathsf{Init}^{\mathsf{RO}}(\tilde{K}, \boldsymbol{K}_i, R, \sigma_i, i)$

return R

Proc. $\textsc{Generate}()$

$R \leftarrow\!\!\$\ \mathcal{R}$

return R

Security notion: RESIST

Game $\text{RESIST}^{\mathscr{B}}_{\boldsymbol{\Pi},\tilde{\boldsymbol{\Pi}}}$

$\overrightarrow{\boldsymbol{K}}_i, \sigma_i \leftarrow \perp$ everywhere

$\tilde{K} \leftarrow\$\ \tilde{\mathcal{K}}$

$(\mathsf{Lk}, \tau) \leftarrow\$\ \mathscr{B}^{\mathsf{RO}}(\tilde{K})$

$b \leftarrow\$\ \{0,1\}$

$b' \leftarrow\$\ \mathscr{B}^{\textsc{Leak},\textsc{Init},\textsc{Enc},\mathsf{RO}}(\tilde{K}, \tau)$

return $(b = b')$

Oracle $\textsc{Leak}(i)$

if $(\boldsymbol{K}_i \neq \perp)$ **then return** $\perp$

$\boldsymbol{K}_i \leftarrow\$\ \mathcal{K}$; $L \leftarrow\$\ \mathsf{Lk}^{\mathsf{RO}}(\boldsymbol{K}_i)$

return L

Oracle $\textsc{Enc}(M, A, i)$

if $(\sigma_i = \perp)$ **then return** $\perp$

$C_0, \sigma_i \leftarrow\$\ \tilde{\boldsymbol{\mathcal{E}}}.\mathsf{Enc}^{\mathsf{RO}}(\tilde{K}, \boldsymbol{K}_i, M, A, \sigma_i, i)$

$C_1 \leftarrow\$\ \{0,1\}^{|C_0|}$

return C_b

Oracle $\textsc{Init}(i)$

if $(\boldsymbol{K}_i = \perp)$ **then return** $\perp$

$R \leftarrow\$\ \textsc{Generate}()$

if $R = \perp$ **then** abort

$\sigma_i \leftarrow\$\ \tilde{\boldsymbol{\mathcal{E}}}.\mathsf{Init}^{\mathsf{RO}}(\tilde{K}, \boldsymbol{K}_i, R, \sigma_i, i)$

return R

Proc. $\textsc{Generate}()$

$R \leftarrow\$\ \mathcal{R}$

return R

Security notion: RESIST

Game $\text{RESIST}^{\mathscr{B}}_{\mathbf{\Pi},\tilde{\mathbf{\Pi}}}$

$\overrightarrow{\boldsymbol{K}}_i, \sigma_i \leftarrow \perp$ everywhere

$\tilde{K} \leftarrow\$ \tilde{\mathcal{K}}$

$(\mathsf{Lk}, \tau) \leftarrow\$ \mathscr{B}^{\mathsf{RO}}(\tilde{K})$

$b \leftarrow\$ \{0,1\}$

$b' \leftarrow\$ \mathscr{B}^{\textsc{Leak},\textsc{Init},\textsc{Enc},\mathsf{RO}}(\tilde{K}, \tau)$

return $(b = b')$

Oracle $\textsc{Leak}(i)$

if $(\boldsymbol{K}_i \neq \perp)$ **then return** $\perp$

$\boldsymbol{K}_i \leftarrow\$ \boldsymbol{\mathcal{K}}$; $L \leftarrow\$ \mathsf{Lk}^{\mathsf{RO}}(\boldsymbol{K}_i)$

return L

Oracle $\textsc{Enc}(M, A, i)$

if $(\sigma_i = \perp)$ **then return** $\perp$

$C_0, \sigma_i \leftarrow\$ \tilde{\boldsymbol{\mathcal{E}}}.\mathsf{Enc}^{\mathsf{RO}}(\tilde{K}, \boldsymbol{K}_i, M, A, \sigma_i, i)$

$C_1 \leftarrow\$ \{0,1\}^{|C_0|}$

return C_b

Oracle $\textsc{Init}(i)$

if $(\boldsymbol{K}_i = \perp)$ **then return** $\perp$

$R \leftarrow\$ \textsc{Generate}()$

if $R = \perp$ **then** abort

$\sigma_i \leftarrow\$ \tilde{\boldsymbol{\mathcal{E}}}.\mathsf{Init}^{\mathsf{RO}}(\tilde{K}, \boldsymbol{K}_i, R, \sigma_i, i)$

return R

Proc. $\textsc{Generate}()$

$R \leftarrow\$ \mathcal{R}$

return R

Security notion: RESIST

Game $\mathrm{RESIST}^{\mathscr{B}}_{\mathbf{\Pi},\tilde{\mathbf{\Pi}}}$

$\overrightarrow{\boldsymbol{K}}_i, \sigma_i \leftarrow \perp$ everywhere

$\tilde{K} \leftarrow\!\!\$\ \tilde{\mathcal{K}}$

$(\mathsf{Lk}, \tau) \leftarrow\!\!\$\ \mathscr{B}^{\mathsf{RO}}(\tilde{K})$

$b \leftarrow\!\!\$\ \{0,1\}$

$b' \leftarrow\!\!\$\ \mathscr{B}^{\textsc{Leak},\textsc{Init},\textsc{Enc},\mathsf{RO}}(\tilde{K}, \tau)$

return $(b = b')$

Oracle $\textsc{Leak}(i)$

if $(\boldsymbol{K}_i \neq \perp)$ **then return** $\perp$

$\boldsymbol{K}_i \leftarrow\!\!\$\ \mathcal{K};\ L \leftarrow\!\!\$\ \mathsf{Lk}^{\mathsf{RO}}(\boldsymbol{K}_i)$

return L

Oracle $\textsc{Enc}(M, A, i)$

if $(\sigma_i = \perp)$ **then return** $\perp$

$C_0, \sigma_i \leftarrow\!\!\$\ \tilde{\boldsymbol{\mathcal{E}}}.\mathsf{Enc}^{\mathsf{RO}}(\tilde{K}, \boldsymbol{K}_i, M, A, \sigma_i, i)$

$C_1 \leftarrow\!\!\$\ \{0,1\}^{|C_0|}$

return C_b

Oracle $\textsc{Init}(i)$

if $(\boldsymbol{K}_i = \perp)$ **then return** $\perp$

$R \leftarrow\!\!\$\ \textsc{Generate}()$

if $R = \perp$ **then** abort

$\sigma_i \leftarrow\!\!\$\ \tilde{\boldsymbol{\mathcal{E}}}.\mathsf{Init}^{\mathsf{RO}}(\tilde{K}, \boldsymbol{K}_i, R, \sigma_i, i)$

return R

Proc. $\textsc{Generate}()$

$R \leftarrow\!\!\$\ \mathcal{R}$

return R

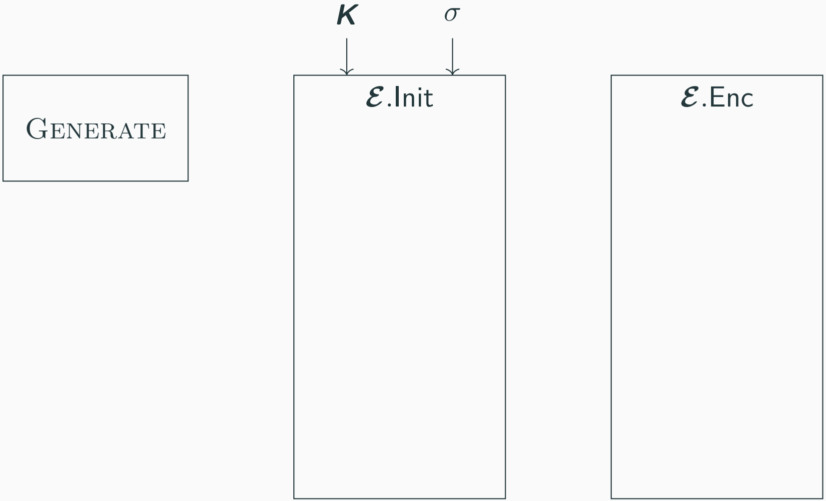
Construction SES$[\Pi, k, P]$**: encryption**

GENERATE

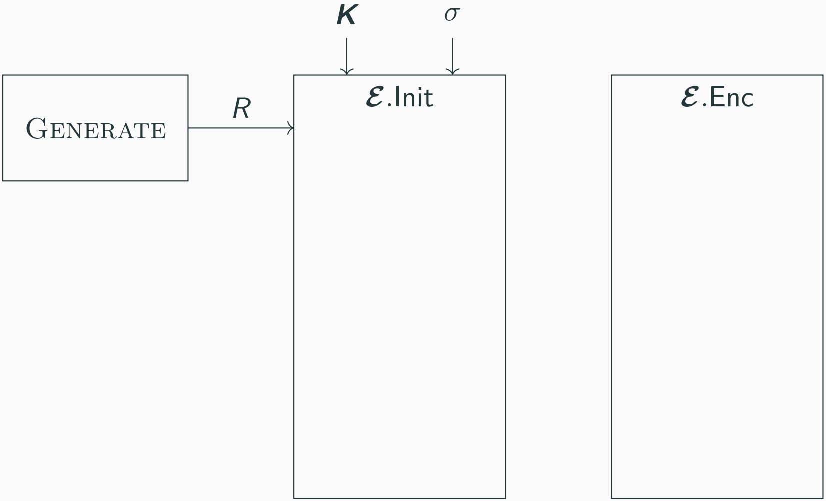
$\mathcal{E}$.Init

$\mathcal{E}$.Enc

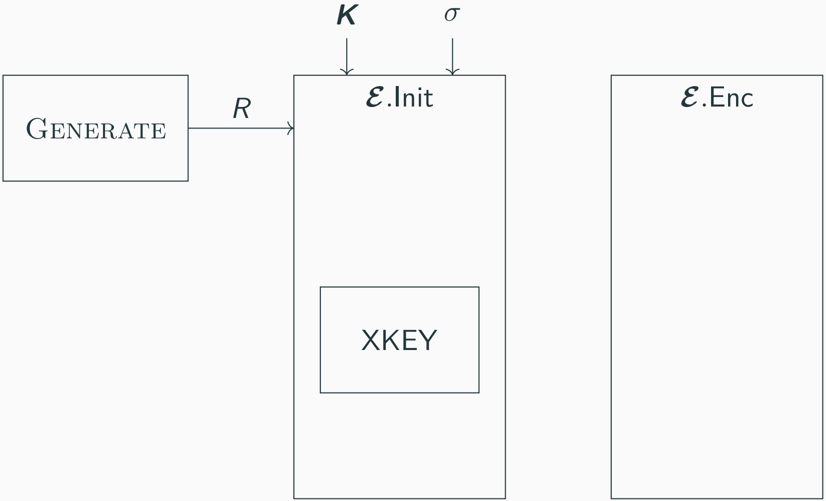
Construction $\mathsf{SES}[\Pi, k, P]$: encryption



Construction $\mathsf{SES}[\Pi, k, P]$**: encryption**

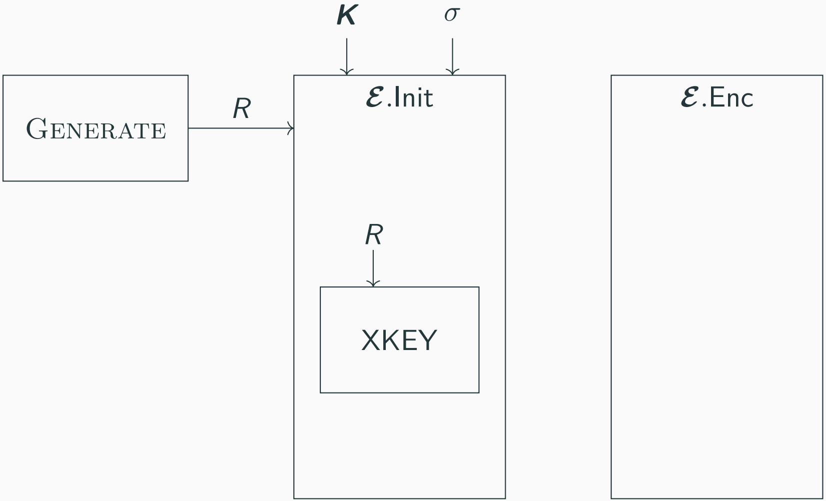


Construction SES[Π, k, P]: encryption



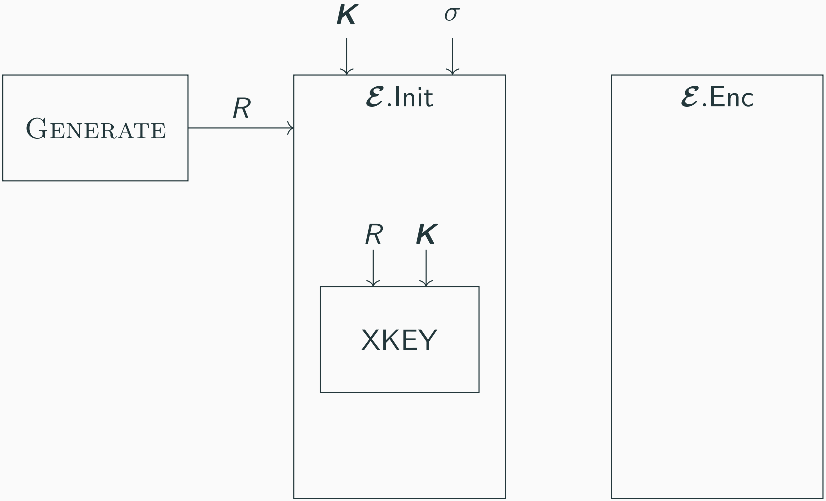
XKEY is the big-key encapsulation scheme from [BKR16].

Construction $\mathsf{SES}[\Pi, k, P]$: encryption



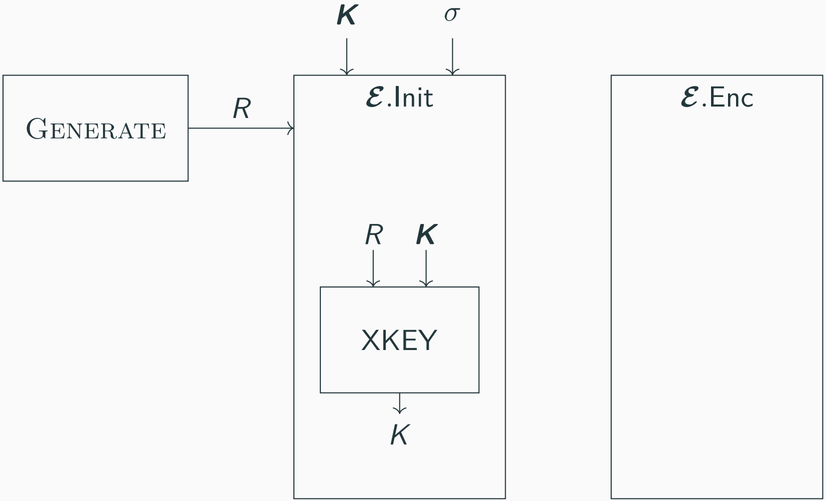
XKEY is the big-key encapsulation scheme from [BKR16].

Construction SES$[\Pi, k, P]$**: encryption**



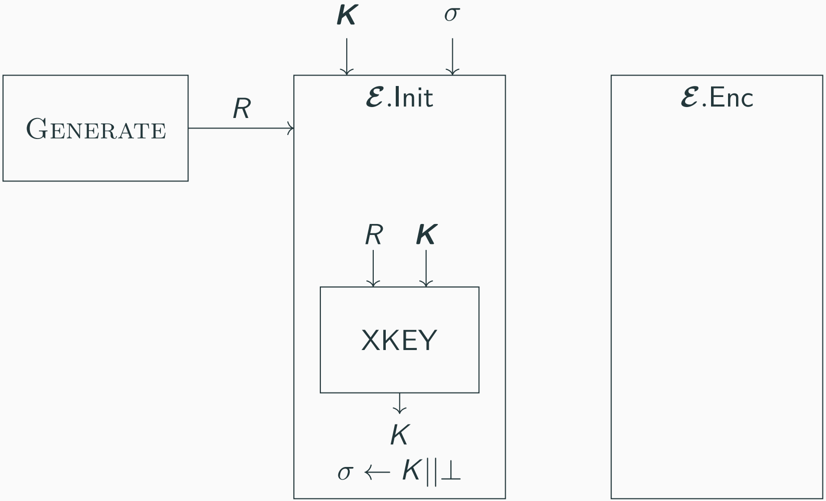
XKEY is the big-key encapsulation scheme from [BKR16].

Construction SES[Π, k, P]: encryption



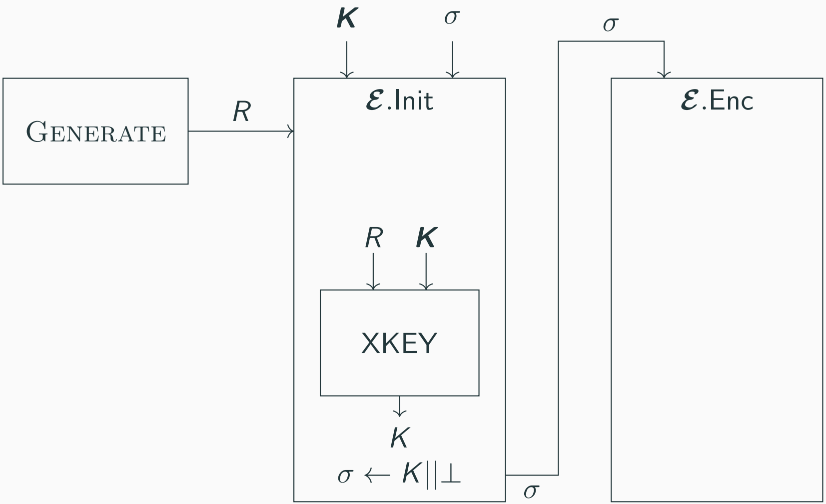
XKEY is the big-key encapsulation scheme from [BKR16].

Construction SES$[\Pi, k, P]$**: encryption**



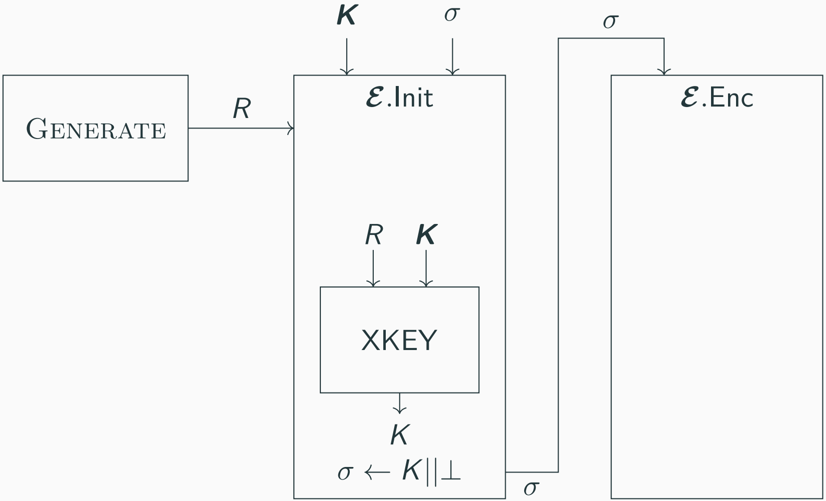
XKEY is the big-key encapsulation scheme from [BKR16].

Construction SES[Π, k, P]: encryption



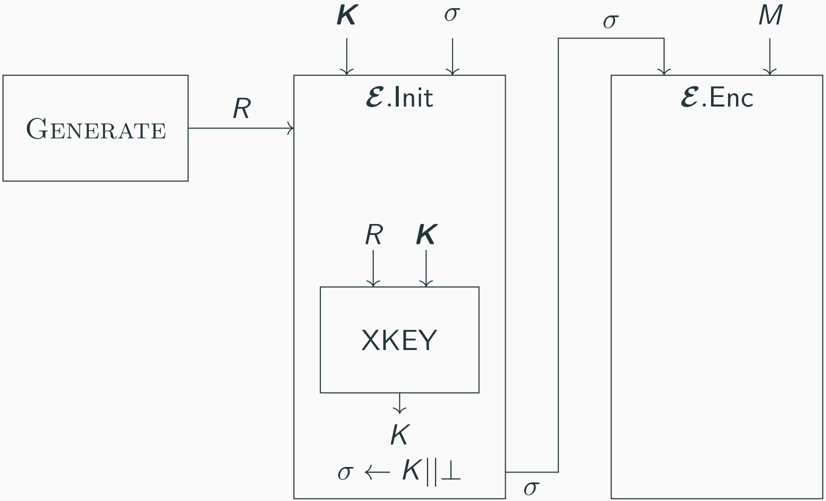
XKEY is the big-key encapsulation scheme from [BKR16].

Construction $\mathsf{SES}[\Pi, k, P]$: encryption



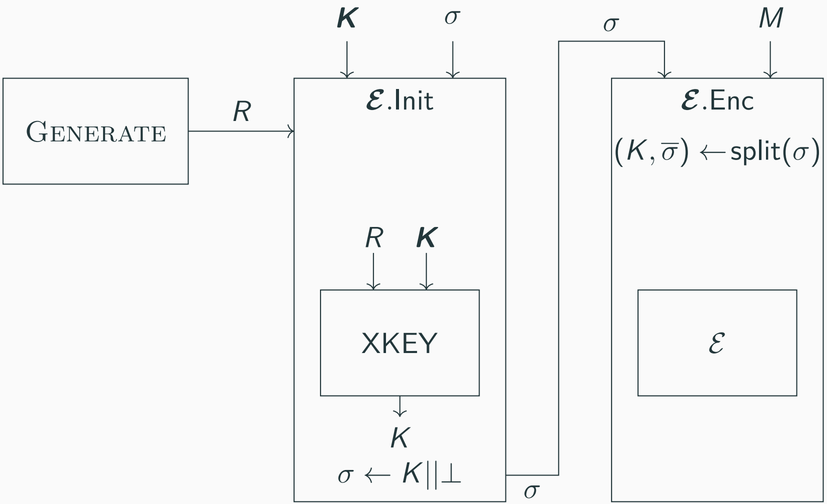
XKEY is the big-key encapsulation scheme from [BKR16].

Construction SES[Π, k, P]: encryption



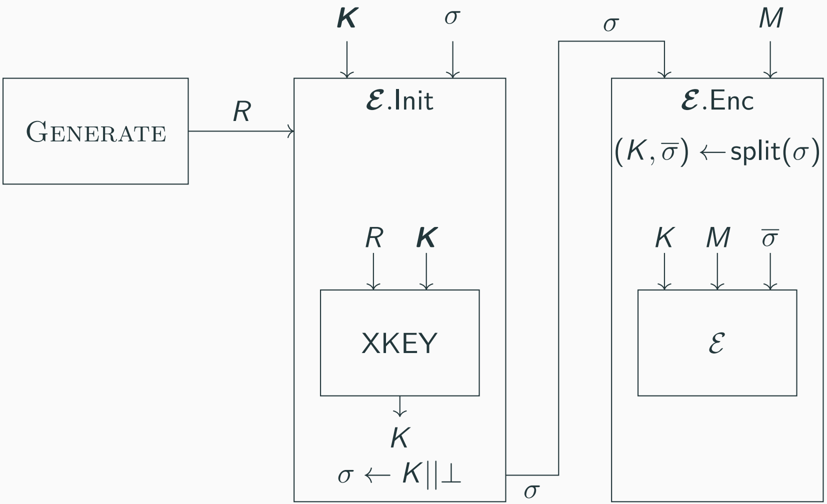
XKEY is the big-key encapsulation scheme from [BKR16].

Construction SES[Π, k, P]**: encryption**



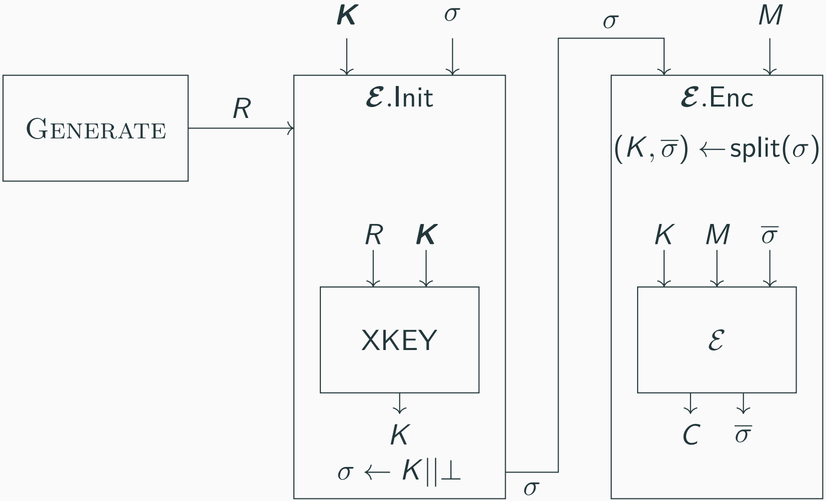
XKEY is the big-key encapsulation scheme from [BKR16].

Construction SES[Π, k, P]**: encryption**



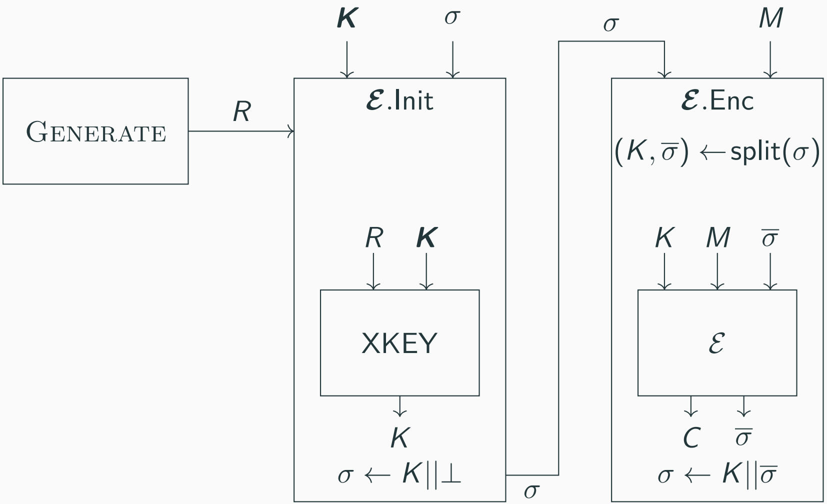
XKEY is the big-key encapsulation scheme from [BKR16].

Construction SES[Π, k, P]: encryption



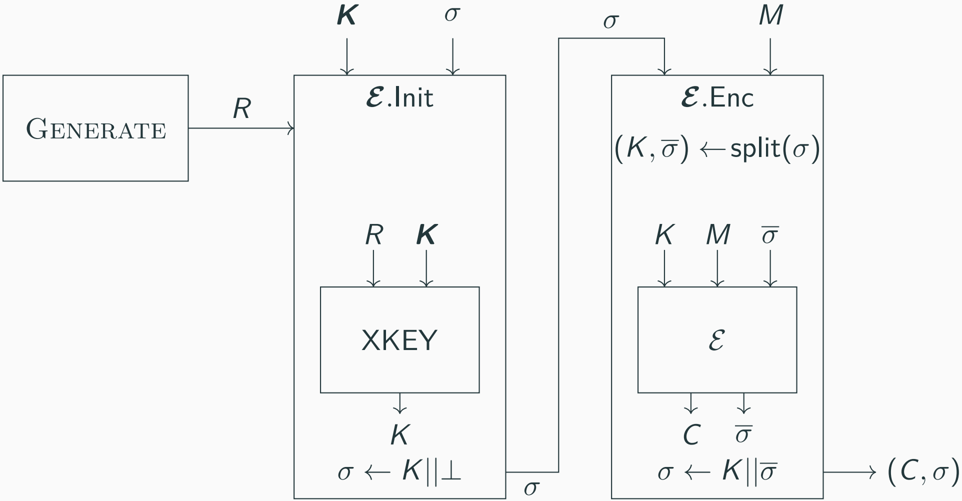
XKEY is the big-key encapsulation scheme from [BKR16].

Construction $\mathsf{SES}[\Pi, k, P]$: encryption



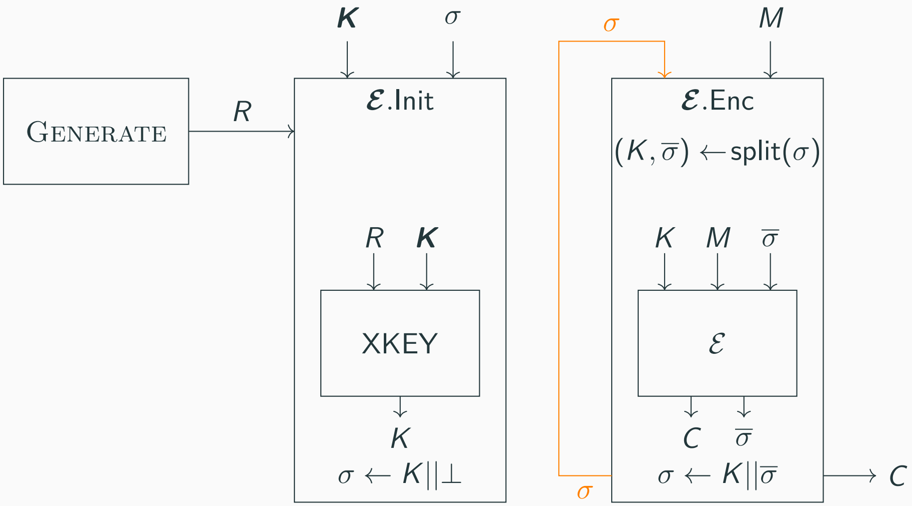
XKEY is the big-key encapsulation scheme from [BKR16].

Construction SES[Π, k, P]: encryption



XKEY is the big-key encapsulation scheme from [BKR16].

Construction SES[Π, k, P]: encryption



XKEY is the big-key encapsulation scheme from [BKR16].

Big Brother is defeated

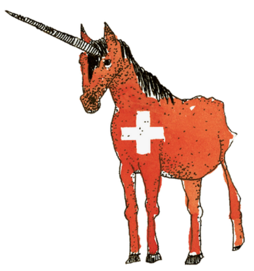
Theorem

Let $\Pi = (\mathcal{K}, \mathcal{E}, \mathcal{D})$ *be an encryption scheme with unique ciphertexts and* $\mathcal{K} = \{0,1\}^\kappa$*. Let* k*,* P*,* h *be positive integers. Let* $\mathbf{\Pi} = \mathrm{SES}[\Pi, k, P]$ *and let* $\tilde{\mathbf{\Pi}} = (\tilde{\mathcal{K}}, \tilde{\boldsymbol{\mathcal{E}}}, \tilde{\mathcal{D}})$ *be a subversion of* $\mathbf{\Pi}$ *that meets the decryptability condition. Let* $\mathscr{B}$ *be an adversary. Then*

$$\mathbf{Adv}^{\mathsf{resist}}_{\mathbf{\Pi}, \tilde{\mathbf{\Pi}}, h, p, \ell}(\mathscr{B}) \le \Delta_1 + \Delta_2 + \Delta_3,$$

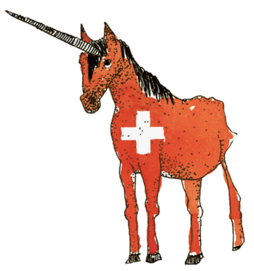
with $\Delta_1 = 0$*,* $\Delta_2 = 2 \cdot q_K \cdot \mathbf{Adv}^{\mathsf{ukey}}_{\mathsf{XKEY}}(\mathscr{A})$*,* $\Delta_3 = 2 \cdot q_I \cdot \mathbf{Adv}^{\mathsf{ind\$}}_{\Pi}(\mathscr{A}')$*.*

How to get secure randomness



We show how to instantiate GENERATE with the unicorn protocol [LW17]:

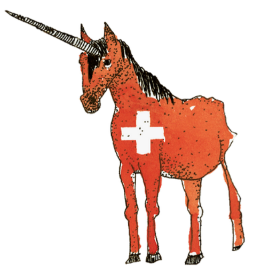
How to get secure randomness



We show how to instantiate GENERATE with the unicorn protocol [LW17]:

- provably uncontestable randomness generation;

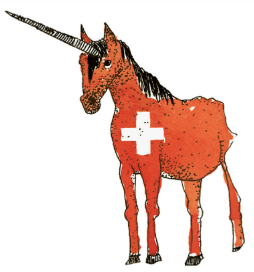
How to get secure randomness



We show how to instantiate GENERATE with the unicorn protocol [LW17]:

- provably uncontestable randomness generation;
- interactive protocol for joint random sampling by any number of parties;

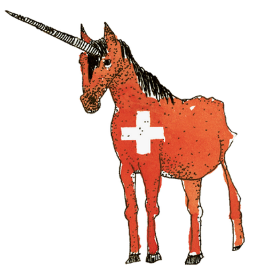
How to get secure randomness



We show how to instantiate GENERATE with the unicorn protocol [LW17]:

- provably uncontestable randomness generation;
- interactive protocol for joint random sampling by any number of parties;
- each participant can verify that no tampering occurred, without trusting others.

How to get secure randomness



We show how to instantiate GENERATE with the unicorn protocol [LW17]:

- provably uncontestable randomness generation;
- interactive protocol for joint random sampling by any number of parties;
- each participant can verify that no tampering occurred, without trusting others.

We prove security of our construction with unicorn, by bounding $\mathbf{Adv}^{\mathsf{ukey}}_{\mathsf{XKEY}}$.

Conclusion and future work

Conclusion and future work

Coordinated ASAs and key exfiltration attacks break standalone symmetric encryption: relying on secure external randomness restores security against both attack vectors.

Conclusion and future work

Coordinated ASAs and key exfiltration attacks break standalone symmetric encryption: relying on secure external randomness restores security against both attack vectors.

Future work:

Conclusion and future work

Coordinated ASAs and key exfiltration attacks break standalone symmetric encryption: relying on secure external randomness restores security against both attack vectors.

Future work:

- more realistic leakage models;

Conclusion and future work

Coordinated ASAs and key exfiltration attacks break standalone symmetric encryption: relying on secure external randomness restores security against both attack vectors.

Future work:

- more realistic leakage models;
- relaxing decryptability for stronger subversions;

Conclusion and future work

Coordinated ASAs and key exfiltration attacks break standalone symmetric encryption: relying on secure external randomness restores security against both attack vectors.

Future work:

- more realistic leakage models;
- relaxing decryptability for stronger subversions;
- faster sources of secure randomness (VDF-based or distributed random beacons);

Conclusion and future work

Coordinated ASAs and key exfiltration attacks break standalone symmetric encryption: relying on secure external randomness restores security against both attack vectors.

Future work:

- more realistic leakage models;
- relaxing decryptability for stronger subversions;
- faster sources of secure randomness (VDF-based or distributed random beacons);
- consider advantages of secure randomness more generally.

Mihir Bellare, Daniel Kane, and Phillip Rogaway.
Big-key symmetric encryption: Resisting key exfiltration.
In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 373–402. Springer, Berlin, Heidelberg, August 2016.

Mihir Bellare, Kenneth G. Paterson, and Phillip Rogaway.
Security of symmetric encryption against mass surveillance.
In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 1–19. Springer, Berlin, Heidelberg, August 2014.

Arjen K. Lenstra and Benjamin Wesolowski.
Trustworthy public randomness with sloth, unicorn, and trx.
Int. J. Appl. Cryptogr., 3(4):330–343, 2017.