UC Mixnets

Myrto Arapinis and Margot Catinaud

June 2025

1 Introduction

2 A UC framework for mixnets

2.1 A game-based secure mixnet

2.1.1 Mixnet definition

Definition 1 (Mixnet protocol)**.** *A* mixnet protocol Π_{MN} *is a tuple of algorithms*

$$\Pi_{\mathsf{MN}} = \big(\textbf{Setup}, \textbf{Prepare}, \textbf{Mix}, \textbf{Verify}, \textbf{Recover}, \textbf{Decrypt}\big)$$

where

- **Setup** *takes as input a security parameter* $\eta \in \mathbb{N}^*$ *and two natural numbers* $n, p \in \mathbb{N}^*$*. Informally,* n *is the number of* senders $(\mathsf{S}_i)_{i=1}^n$ *while* p *is the number of* mix-servers $(\mathsf{M}_j)_{j=1}^p$*. Then,* **Setup** *outputs a public parameter* σ_{pub} *of the mixnet protocol and* p *private parameters* $\big(\sigma_{\mathsf{priv}}^{(j)}\big)_{j=1}^p$ *for each mix-server ;*

- **Prepare** *takes as input an identifier* $\mathtt{id} \in [\![1; n]\!]$ *of a sender* $\mathsf{S}_{\mathtt{id}}$*, the public parameter* σ_{pub}*, with* $pk \in \sigma_{\mathsf{pub}}$*, and a message* $m \in \mathcal{M}$*. Then,* **Prepare** *generates a random value* $s \xleftarrow{\$} \mathcal{R}_{\mathbb{CS}}$ *uniformly at random. Finally,* **Prepare** *outputs an encrypted packet* $\mathfrak{p} \overset{def}{=} (c, \mathsf{tr})$ *with* $c \in \mathcal{C}_{\mathbb{CS}}$ *a ciphertext and* tr *a proof transcript such that*

$$c \overset{def}{=} \mathsf{Enc}_{\mathbb{CS}}\big(pk, m\,;\, s\big) \qquad \textit{and} \qquad \mathsf{tr} \vDash_{\mathbb{ZK}} \big(\sigma_{\mathsf{pub}}, c, (m, s)\big) \in \mathcal{R}_{\mathbb{CS}}^{\mathsf{crypt}};$$

- **Mix** *takes as input a private parameter* $\sigma_{\mathsf{priv}}^{(\mathtt{id})}$ *for mix-server* $\mathsf{M}_{\mathtt{id}}$*, the public parameter* σ_{pub}*, with* $pk \in \sigma_{\mathsf{pub}}$ *the public key and* $ck \in \sigma_{\mathsf{pub}}$ *a commitment key parameter, and a list of ciphertexts* $\mathbf{c} = (c_i)_{i=1}^n \in \mathcal{C}_{\mathbb{CS}}^n$*. Then,* **Mix** *generates a permutation* $\pi \xleftarrow{\$} \mathfrak{S}_n$ *and two vectors of random values* $\mathbf{r} = (r_i)_{i=1}^n \xleftarrow{\$} \mathcal{R}_{\mathbb{CS}}^n$ *and* $\mathbf{s} = (s_i)_{i=1}^p \xleftarrow{\$} \mathcal{R}_{\mathbb{KS}[\mathfrak{S}_n]}$ *uniformly at random. Finally,* **Mix** *outputs*

  (i) a new list of ciphertexts $\mathbf{c}' = (c'_i)_{i=1}^n \in \mathcal{C}_{\mathbb{CS}}^n$ *such that*

$$\forall\, i \in [\![1; n]\!],\ c'_i \overset{def}{=} \phi_{\mathbb{CS}}\Big(pk, \sigma_{\mathsf{priv}}^{(\mathtt{id})}, c_{\pi(i)}\,;\, r_i\Big)\,;$$

  (ii) a commitment to the permutation π *:* $\mathbf{a} \overset{def}{=} \mathsf{Com}_{\mathfrak{S}_n}(ck, \pi\,;\, \mathbf{s}) \in \mathcal{K}_{\mathfrak{S}_n}$ *;*

  (iii) and a proof transcript tr *such that*

$$\mathsf{tr} \vDash_{\mathbb{ZK}} \begin{cases} & \Big(\big(\sigma_{\mathsf{pub}} \setminus \{pk\}\big), pk, \sigma_{\mathsf{priv}}^{(\mathtt{id})}\Big) \in \mathcal{R}_{\mathbb{CS}}^{\mathsf{skey}} \\ \wedge & \Big(\sigma_{\mathsf{pub}}, (\mathbf{a}, \mathbf{c}, \mathbf{c}'), \big(\pi, \mathbf{s}, \sigma_{\mathsf{priv}}^{(\mathtt{id})}, \mathbf{r}\big)\Big) \in \mathcal{R}_{\mathbb{CS},\,\mathfrak{S}_n}^{\mathsf{shuffle}} \end{cases}. \tag{Φ}$$

- **Verify** *takes as input the public parameter* σ_{pub} *and two paquet sets* $\mathfrak{P}_{\mathtt{id}}$ *and* $\mathfrak{P}_{\mathtt{id}'}$ *where* $\mathtt{id}, \mathtt{id}' \in [\![1; n]\!]$. *Informally, we suppose that the paquet set* $\mathfrak{P}_{\mathtt{id}}$ *is already checked as valid and we want to verify that the other paquet set* $\mathfrak{P}_{\mathtt{id}'}$ *is also valid. Then,* **Verify** *works as follows:*

  (1) firstly, it extract from paquet set $\mathfrak{P}_{\mathtt{id}'}$ *a list of ciphertexts* $\mathbf{c}' \in \mathcal{C}_{\mathbb{CS}}^n$*, a commitment value* $\mathbf{a} \in \mathcal{K}_{\mathfrak{S}_n}$*, and a proof transcript* tr' *;*

  (2) secondly, it extract from paquet set $\mathfrak{P}_{\mathtt{id}}$ *a list of ciphertexts* $\mathbf{c} \in \mathcal{C}_{\mathbb{CS}}^n$ *;*

  (3) and finally, **Verify** *outputs* 1 *if and only if the following property holds*

$$\mathsf{tr}' \vDash_{\mathbb{ZK}} \left(pk \in \mathcal{L}_{\sigma_{\mathsf{pub}} \setminus \{pk\}} \left(\mathcal{R}_{\mathbb{CS}}^{\mathsf{skey}} \right) \right) \ \wedge \ \left((\mathbf{a}, \mathbf{c}, \mathbf{c}') \in \mathcal{L}_{\sigma_{\mathsf{pub}}} \left(\mathcal{R}_{\mathbb{CS}, \mathfrak{S}_n}^{\mathsf{shuffle}} \right) \right) \tag{Θ}$$

-

2.1.2 Common oracles

$\mathcal{O}\mathsf{DishonestSend}(\mathtt{id}, \mathfrak{p})$ —
if $(\mathtt{id} \notin \mathcal{I}_{\mathsf{S}})$ **then**
  $\mathfrak{P}_0 \leftarrow \mathfrak{P}_0 \,||\, \mathfrak{p}$;
  $\mathcal{I}_{\mathsf{S}} \leftarrow \mathcal{I}_{\mathsf{S}} \cup \{\mathtt{id}\}$.

$\mathcal{O}\mathsf{HonestMix}(\mathtt{id})$ —
$(\mathbf{c}, \overrightarrow{\mathsf{tr}}) \leftarrow$ **Recover** $\left(\sigma_{\mathsf{pub}}, (\sigma_{\mathsf{priv}}^{(j)})_{j \in \mathcal{I}_{\mathsf{M}}}, (\mathfrak{P}_j)_{j \in \mathcal{I}_{\mathsf{M}}} \right)$;
$(\mathbf{c}', \overrightarrow{\mathsf{tr}}') \leftarrow$ **Mix** $\left(\sigma_{\mathsf{priv}}^{(\mathtt{id})}, \sigma_{\mathsf{pub}}, \mathbf{c} \right)$;
$\mathcal{I}_{\mathsf{M}} \leftarrow \mathcal{I}_{\mathsf{M}} \cup \{\mathtt{id}\}$;
$\mathfrak{P}_{\mathtt{id}} \leftarrow$ **CreateBB** $\left(\mathbf{c}', \overrightarrow{\mathsf{tr}} \circledcirc \overrightarrow{\mathsf{tr}}' \right)$.

$\mathcal{O}\mathsf{DishonestMix}(\mathtt{id}, \mathbf{c}, \overrightarrow{\mathsf{tr}})$ —
$\mathcal{I}_{\mathsf{M}} \leftarrow \mathcal{I}_{\mathsf{M}} \cup \{\mathtt{id}\}$;
$\mathfrak{P}_{\mathtt{id}} \leftarrow$ **CreateBB** $\left(\mathbf{c}, \overrightarrow{\mathsf{tr}} \right)$.

$\mathcal{O}\mathsf{Open}()$ —
if $\left(M_0 \stackrel{+}{=} M_1 \right)$ **then**
  $(\mathbf{c}, \_) \leftarrow$ **Recover** $\left((\sigma_{\mathsf{priv}}^{(j)})_{j=1}^{p}, (\mathfrak{P}_i)_{i=0}^{p} \right)$;
  $\mathbf{m}' \leftarrow$ **Decrypt** $\left((\sigma_{\mathsf{priv}}^{(j)})_{j=1}^{p}, \mathbf{c} \right)$;
  return $\mathbf{m}'$;
else return $\perp$.

$\mathcal{O}\mathsf{PublishBB}(\mathtt{id})$ —
return $\mathfrak{P}_{\mathtt{id}}$.

2.1.3 Mixnet privacy

$\mathcal{O}\mathsf{HonestSend}^{(\beta)}(\mathtt{id}, m_0, m_1)$ —
if $(\mathtt{id} \notin \mathcal{I}_{\mathsf{S}})$ **then**
  $M_0 \leftarrow m_0 \uplus M_0$; $M_1 \leftarrow m_1 \uplus M_1$;
  $\mathfrak{p}_\beta \leftarrow$ **Prepare** $\left(\mathtt{id}, \sigma_{\mathsf{pub}}, m_\beta \right)$;
  $\mathfrak{P}_0 \leftarrow \mathfrak{P}_0 \,||\, \mathfrak{p}_\beta$;
  $\mathcal{I}_{\mathsf{S}} \leftarrow \mathcal{I}_{\mathsf{S}} \cup \{\mathtt{id}\}$.

2.1.4 Mixnet integrity

$\mathcal{O}\mathsf{HonestSend}(\mathtt{id}, m)$ —
if $(\mathtt{id} \notin \mathcal{I}_P)$ **then**
  $M \leftarrow m \uplus M$;
  $\mathfrak{p} \leftarrow$ **Prepare** $\left(\mathtt{id}, \sigma_{\mathsf{pub}}, m \right)$;
  $\mathfrak{P}_0 \leftarrow \mathfrak{P}_0 \,||\, \mathfrak{p}$;
  $\mathcal{I}_P \leftarrow \mathcal{I}_P \cup \{\mathtt{id}\}$.

Game 1: Cryptographic game $\mathsf{Privacy}^{\mathcal{A},\beta}_{\Pi_{\mathsf{MN}}}(\eta)$ of privacy for a mixnet protocol Π_{MN}

Configuration phase

$M_0, M_1 \leftarrow \{\!\{\}\!\}$;
$\mathcal{I}_{\mathsf{S}}, \mathcal{I}_{\mathsf{M}} \leftarrow \varnothing$;
$(n, p, \mathcal{H}_{\mathsf{S}}, \mathcal{H}_{\mathsf{M}}) \leftarrow \mathcal{A}()$; // $\mathcal{A}$ chooses n senders and p mix-servers
$(\sigma^{(1)}_{\mathsf{priv}}, \ldots, \sigma^{(p)}_{\mathsf{priv}}, \sigma_{\mathsf{pub}}) \leftarrow$ **Setup** (η) ;
$\mathfrak{P}_0 \leftarrow (\sigma^{(j)}_{\mathsf{priv}})_{j \notin \mathcal{H}_{\mathsf{M}}} \,||\, \sigma_{\mathsf{pub}} \,||$ **nil** ;

Sending phase

The adversary $\mathcal{A}$ has access to oracles $\mathcal{O}\mathsf{HonestSend}$ and $\mathcal{O}\mathsf{DishonestSend}$ as follows.
Do
  $\mathtt{id} \leftarrow \mathcal{A}(\mathcal{I}_{\mathsf{S}})$; // *Suppose the adversary $\mathcal{A}$ wants to run the sender $\mathsf{S}_{\mathtt{id}}$ for $\mathtt{id} \in [\![1; n]\!] \setminus \mathcal{I}_{\mathsf{S}}$, then*
  If $\mathtt{id} \in \mathcal{H}_{\mathsf{S}}$ **then** // **Case where $\mathsf{S}_{\mathtt{id}}$ is an honest sender.**
    $\mathcal{A}$ computes two messages m_0, m_1 and calls the oracle $\mathcal{O}\mathsf{HonestSend}^{(\beta)}(\mathtt{id}, m_0, m_1)$.
  Otherwise, // **Case where $\mathsf{S}_{\mathtt{id}}$ is dishonest.**
    $\mathcal{A}$ computes a paquet $\mathtt{p}$ and calls the oracle $\mathcal{O}\mathsf{DishonestSend}(\mathtt{id}, \mathtt{p})$.
While $\mathrm{Card}(\mathcal{I}_{\mathsf{S}}) < n$
At the end of the sending phase (*i.e.* when $\mathrm{Card}(\mathcal{I}_{\mathsf{S}}) = n$), the adversary $\mathcal{A}$ calls the oracle $\mathcal{O}\mathsf{PublishBB}(0)$ and cannot call both oracles $\mathcal{O}\mathsf{HonestSend}$ and $\mathcal{O}\mathsf{DishonestSend}$ anymore.

Mixing phase

if $\left(\neg (M_0 \stackrel{+}{=} M_1)\right)$ **then return** $\perp$;
In a similar way as in the sending phase, the adversary $\mathcal{A}$ has access to oracles $\mathcal{O}\mathsf{HonestMix}$ and $\mathcal{O}\mathsf{DishonestMix}$ whether $\mathtt{id} \in \mathcal{H}_{\mathsf{M}}$ or not. All oracles inputs are computed by the adversary $\mathcal{A}$.
Besides, after each call to mix oracles $\mathcal{O}\mathsf{HonestMix}$ or $\mathcal{O}\mathsf{DishonestMix}$ on identity $\mathtt{id}$, $\mathcal{A}$ calls oracle $\mathcal{O}\mathsf{PublishBB}(\mathtt{id})$.

Conclusion

The adversary $\mathcal{A}$ calls, only once, the oracle $\mathcal{O}\mathsf{Open}()$ to obtain the decrypted messages list $\mathbf{m}$.
$b \leftarrow \mathcal{A}()$;
return $(b = \beta)$.

Game 2: Cryptographic game $\mathsf{Integrity}^{\mathcal{A}}_{\Pi_{\mathsf{MN}}}(\eta)$ of integrity for a mixnet protocol Π_{MN}

Configuration phase

$M \leftarrow \{\!\{\}\!\}$;
$\mathcal{I}_{\mathsf{S}}, \mathcal{I}_{\mathsf{M}} \leftarrow \varnothing$;
$(\sigma^{(1)}_{\mathsf{priv}}, \ldots, \sigma^{(p)}_{\mathsf{priv}}, \sigma_{\mathsf{pub}}) \leftarrow$ **Setup** (η) ;
$(n, p, \mathcal{H}_{\mathsf{S}}, \mathcal{H}_{\mathsf{M}}) \leftarrow \mathcal{A}()$;
$\mathfrak{P}_0 \leftarrow (\sigma^{(j)}_{\mathsf{priv}})_{j \notin \mathcal{H}_{\mathsf{M}}} \,||\, \sigma_{\mathsf{pub}} \,||$ **nil** ;

Sending phase

The adversary $\mathcal{A}$ has access to oracles $\mathcal{O}$HonestSend and $\mathcal{O}$DishonestSend as follows.
Do
  $\mathtt{id} \leftarrow \mathcal{A}(\mathcal{I}_{\mathsf{S}})$; // *Suppose that the adversary $\mathcal{A}$ wants to run the sender $\mathsf{S}_{\mathtt{id}}$ for $\mathtt{id} \in [\![1; n]\!] \setminus \mathcal{I}_{\mathsf{S}}$, then*
  If $\mathtt{id} \in \mathcal{H}_{\mathsf{S}}$ **then** // Case where $\mathsf{S}_{\mathtt{id}}$ is an honest sender.
    $\mathcal{A}$ computes a message m and calls the oracle $\mathcal{O}$HonestSend$(\mathtt{id}, m)$.
  Otherwise, // Case where $\mathsf{S}_{\mathtt{id}}$ is dishonest.
    $\mathcal{A}$ computes a message $\mathtt{p}$ and calls the oracle $\mathcal{O}$DishonestSend$(\mathtt{id}, \mathtt{p})$.
While $\mathrm{Card}(\mathcal{I}_{\mathsf{S}}) < n$
At the end of the sending phase (*i.e.* when $\mathrm{Card}(\mathcal{I}_{\mathsf{S}}) = n$), the adversary $\mathcal{A}$ calls the oracle $\mathcal{O}$PublishBB(0) and cannot call both oracles $\mathcal{O}$HonestSend and $\mathcal{O}$DishonestSend anymore.

Mixing phase

In a similar way as in the sending phase, the adversary $\mathcal{A}$ has access to oracles $\mathcal{O}$HonestMix and $\mathcal{O}$DishonestMix whether $\mathtt{id} \in \mathcal{H}_M$ or not. All oracles inputs are computed by the adversary $\mathcal{A}$.
Besides, after each call to mix oracles $\mathcal{O}$HonestMix or $\mathcal{O}$DishonestMix on identity $\mathtt{id}$, $\mathcal{A}$ calls oracle $\mathcal{O}$PublishBB$(\mathtt{id})$.

Conclusion

The adversary $\mathcal{A}$ calls, only once, the oracle $\mathcal{O}$Open() to obtain the decrypted messages list $\mathbf{m}$.
return $(M \subseteq \mathbf{m})$.

2.2 A UC secure mixnet

Ideal functionality 2.1: Ideal mixnet $\mathcal{F}_{\mathsf{MN}}^{n,p,\theta}$

Let $n, p \in \mathbb{N}$ be two natural numbers. Let $\theta \in]0,1]$ be a threshold parameter. We define *the ideal functionality* $\mathcal{F}_{\mathsf{MN}}^{n,p,\theta}$ *for a* (n,p)*-mixnet with mix threshold* θ, running with p mix-servers $(\mathsf{M}_i)_{i=1}^{p}$, n senders $(\mathsf{S}_i)_{i=1}^{n}$, and ideal adversary $\mathcal{S}$, to be the following protocol.

1. Initialize a list $L = \varnothing$, set $\mathcal{I}_{\mathsf{S}} = \varnothing$ and $\mathcal{I}_{\mathsf{M}} = \varnothing$.
2. Upon receiving $(\mathsf{S}_i, \mathbf{Send}, m_i)$. If $i \notin \mathcal{I}_{\mathsf{S}}$, set $\mathcal{I}_{\mathsf{S}} \leftarrow \mathcal{I}_{\mathsf{S}} \cup \{i\}$, and append m_i to the list L. Then send $(\mathsf{S}_i, \mathbf{Send})$ to $\mathcal{S}$.
3. Upon receiving $(\mathsf{M}_j, \mathbf{Mix})$. Set $\mathcal{I}_{\mathsf{M}} \leftarrow \mathcal{I}_{\mathsf{M}} \cup \{j\}$. If $\mathrm{Card}(\mathcal{I}_{\mathsf{M}}) \geqslant \theta p$, then sort the list L lexicographically to form a list L', and hand $(\mathbf{Mixed}, L')$ to the ideal adversary $\mathcal{S}$ and to all mix-servers $(M_i)_{i=1}^{p}$. Otherwise, hand to $\mathcal{S}$ the list $\big((\mathsf{M}_j, \mathbf{Mix})\big)_{i \in \mathcal{I}_{\mathsf{M}}}$.

2.3 Equivalence between game-based security and UC security

Theorem 1. *Let* Π_{MN} *be a mixnet protocol. Then, we have*

$$\boxed{\Pi_{\mathsf{MN}} \vDash \mathsf{Integrity} \wedge \mathsf{Privacy} \iff \Pi_{\mathsf{MN}} \preccurlyeq_{\mathsf{uc}} \mathcal{F}_{\mathsf{MN}}^{n,p,\theta}}$$