

Parameters: The functionality $\mathcal{F}$ is parameterized by values $\ell, m, \in \mathbb{N}$, $\mathbb{1} \in \mathbb{Z}_{2^m}^\ell$, a distance function $\text{CS.Dist} : \mathbb{Z}_{2^m}^\ell \times \mathbb{Z}_{2^m}^\ell \rightarrow [0, 1]$, a threshold $\tau \in (0, \mathbb{1}^2)$, and leakage functions $L_{\text{dev}}, L_{\text{term}}$ both with description $\mathbb{Z}_m^\ell \times \mathbb{Z}_m^\ell \rightarrow \mathbb{Z}$. It interacts with three parties $\mathcal{D}, \mathcal{SP}$ and $\mathcal{T}$ via the following queries.

Queries:

- **On receiving a query of the form ("Enrollment", $\text{sid}, \vec{u}, \mathcal{SP}$) from $\mathcal{D}$:** Only if sid is unmarked, check if $\|\vec{u}\| = \mathbb{1}$. If the check succeeds, send ("Enrolled", $\text{sid}, \mathcal{D}$) to $\mathcal{SP}$ and $\mathcal{D}$, store $(\text{sid}, \vec{u}, \mathcal{D}, \mathcal{SP})$, and mark sid as "Enrolled". Otherwise, send ("Failed", $\text{sid}, \mathcal{D}$) to $\mathcal{SP}$ and $\mathcal{D}$, and mark sid as "Failed".
- **On receiving a query of the form ("Match", $\text{sid}, \text{ssid}, \mathcal{T}$) from $\mathcal{D}$:** Check whether sid is marked and there exists a record of the form $(\text{sid}, \vec{u}, \mathcal{D}, \mathcal{SP})$. If the check fails, send ("Failed", $\text{sid}, \text{ssid}, \mathcal{D}$) to $\mathcal{T}$ and $\mathcal{D}$. Else:
 - (1) send $(\text{sid}, \text{ssid}, \mathcal{D})$ to $\mathcal{T}$;
 - (2) receive $(\text{sid}, \text{ssid}, \vec{w}, \mathcal{D})$ from $\mathcal{T}$;
 - (3) if $\langle \vec{u}, \vec{w} \rangle \geq \tau$, (that is, $\text{CS.Dist}(\vec{u}, \vec{w}) \geq \frac{\tau}{\mathbb{1}^2}$) send $(\text{sid}, \text{ssid}, 1)$ to $\mathcal{D}$ and $\mathcal{T}$, else send $(\text{sid}, \text{ssid}, 0)$. Also send $L_{\text{dev}}(\vec{u}, \vec{w})$ to $\mathcal{D}$ and $L_{\text{term}}(\vec{u}, \vec{w})$ to $\mathcal{T}$.

FIG. 1

2/16

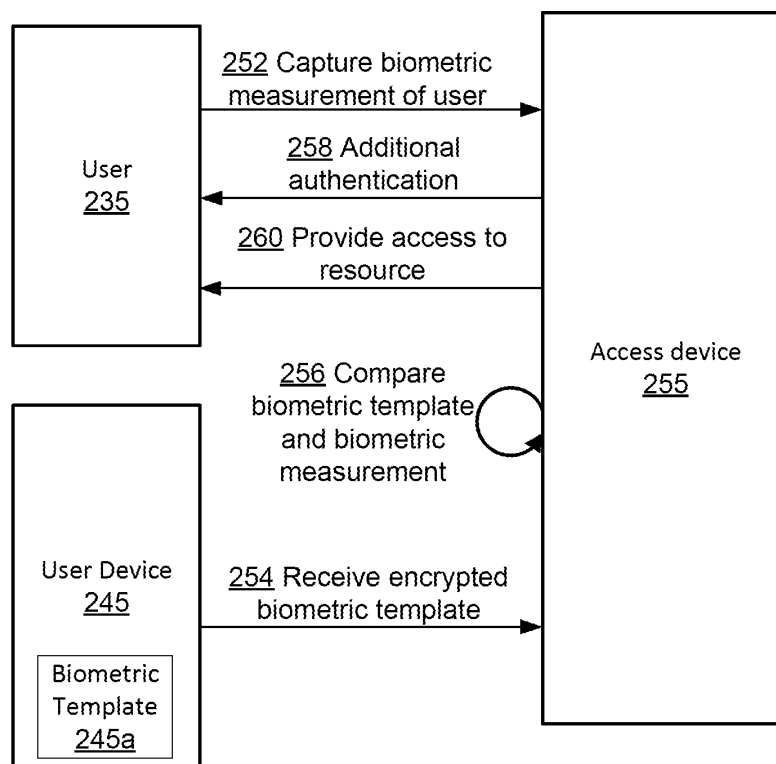


FIG. 2

3/16



FIG. 3

4/16

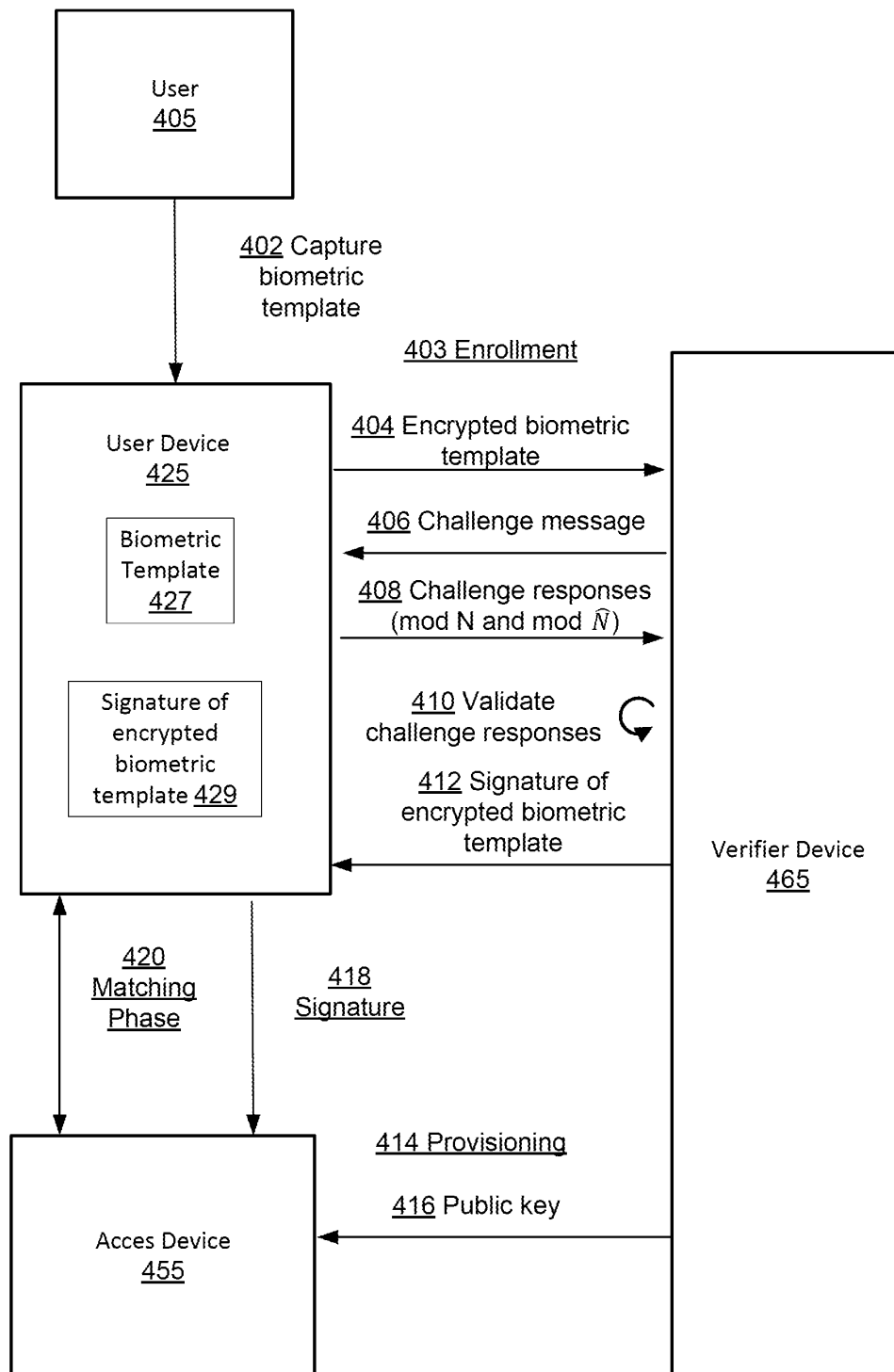


FIG. 4

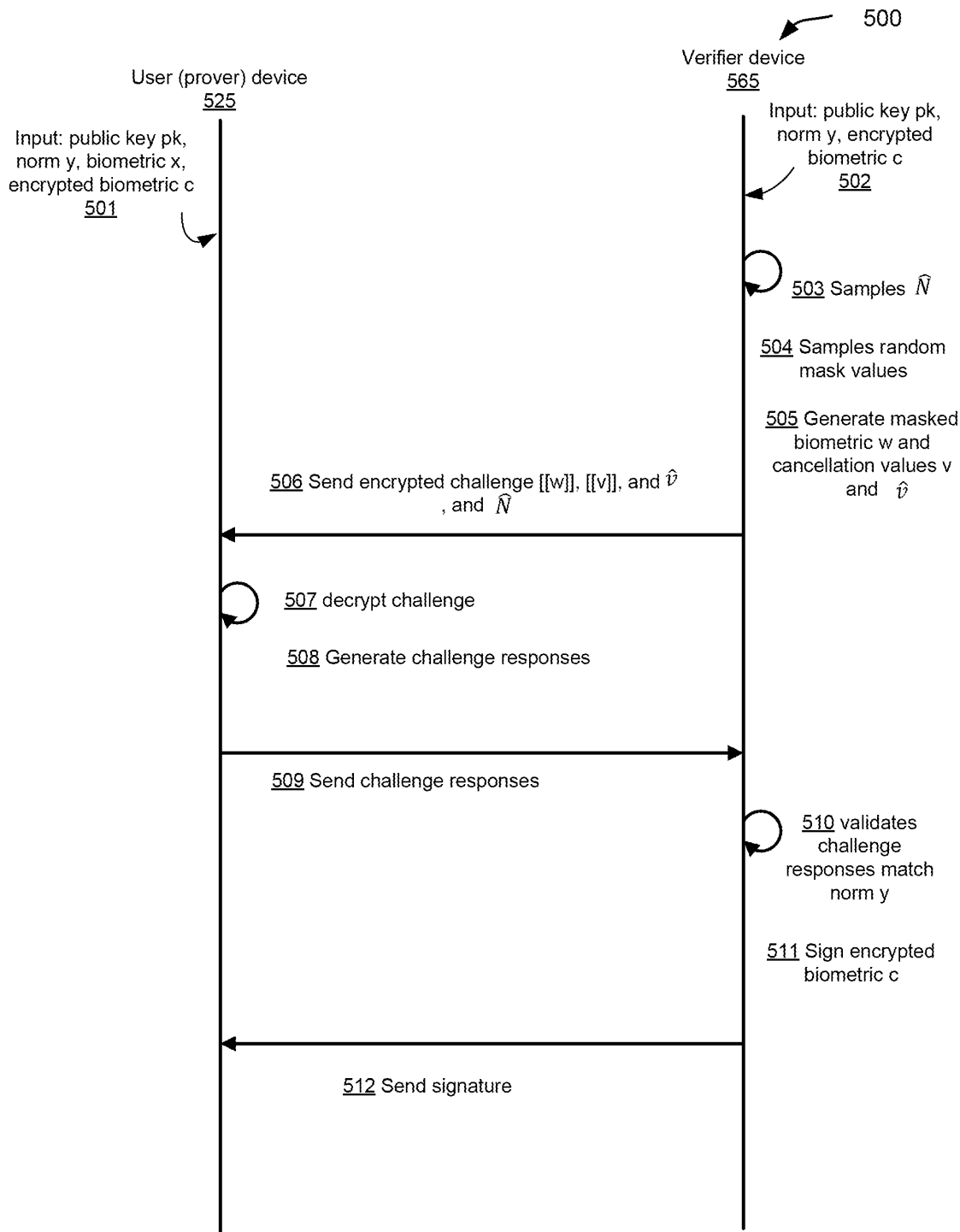


FIG. 5

6/16

Parameters: A Paillier public key $pk = (N, g)$, a value $y \in \mathbb{Z}$, statement $st = (c_1 = \llbracket x_1; r_1 \rrbracket, \dots, c_n = \llbracket x_n; r_n \rrbracket)$, and witness $wit = ((x_1, r_1), \dots, (x_n, r_n))$.

Protocol:

- 610 • V uniformly samples a prime $\hat{N} < N$.
- 620 • V samples random $\alpha, \leftarrow \mathbb{Z}_{2\lambda}, \rho_1, \dots, \rho_n, \beta, \hat{\beta} \leftarrow \mathbb{Z}_N$. Then it sends P the following:
 - 630 – $\llbracket w_i \rrbracket \leftarrow \alpha \llbracket x_i \rrbracket + \llbracket \rho_i \rrbracket$ for all $i \in [n]$,
 - $\llbracket v \rrbracket \leftarrow (\sum_{i \in [n]} (-2\alpha \rho_i \llbracket x_i \rrbracket - \rho_i^2)) + \llbracket \beta \rrbracket$,
 - $\llbracket \hat{v} \rrbracket \leftarrow (\sum_{i \in [n]} c_i \llbracket x_i \rrbracket) + d + \llbracket \hat{\beta} \rrbracket$ where $c_i := (-2\alpha \rho_i \bmod \hat{N}), d := (\sum_{i \in [n]} -\rho_i^2 \bmod \hat{N})$,
 - $\hat{N}$.
- 640 • P decrypts $\llbracket w_i \rrbracket, \llbracket v \rrbracket, \llbracket \hat{v} \rrbracket$ and sends the following to V :
 - 650 – $z := v + \sum_{i \in [n]} w_i^2 \bmod N$,
 - $\hat{z} := \hat{v} + \sum_{i \in [n]} w_i^2 \bmod \hat{N}$.
- 660 • V outputs accept if $z = y\alpha^2 + \beta$ and $\hat{z} \equiv_{\hat{N}} y\alpha^2 + \hat{\beta}$. Otherwise V outputs reject.

FIG. 6

7/16

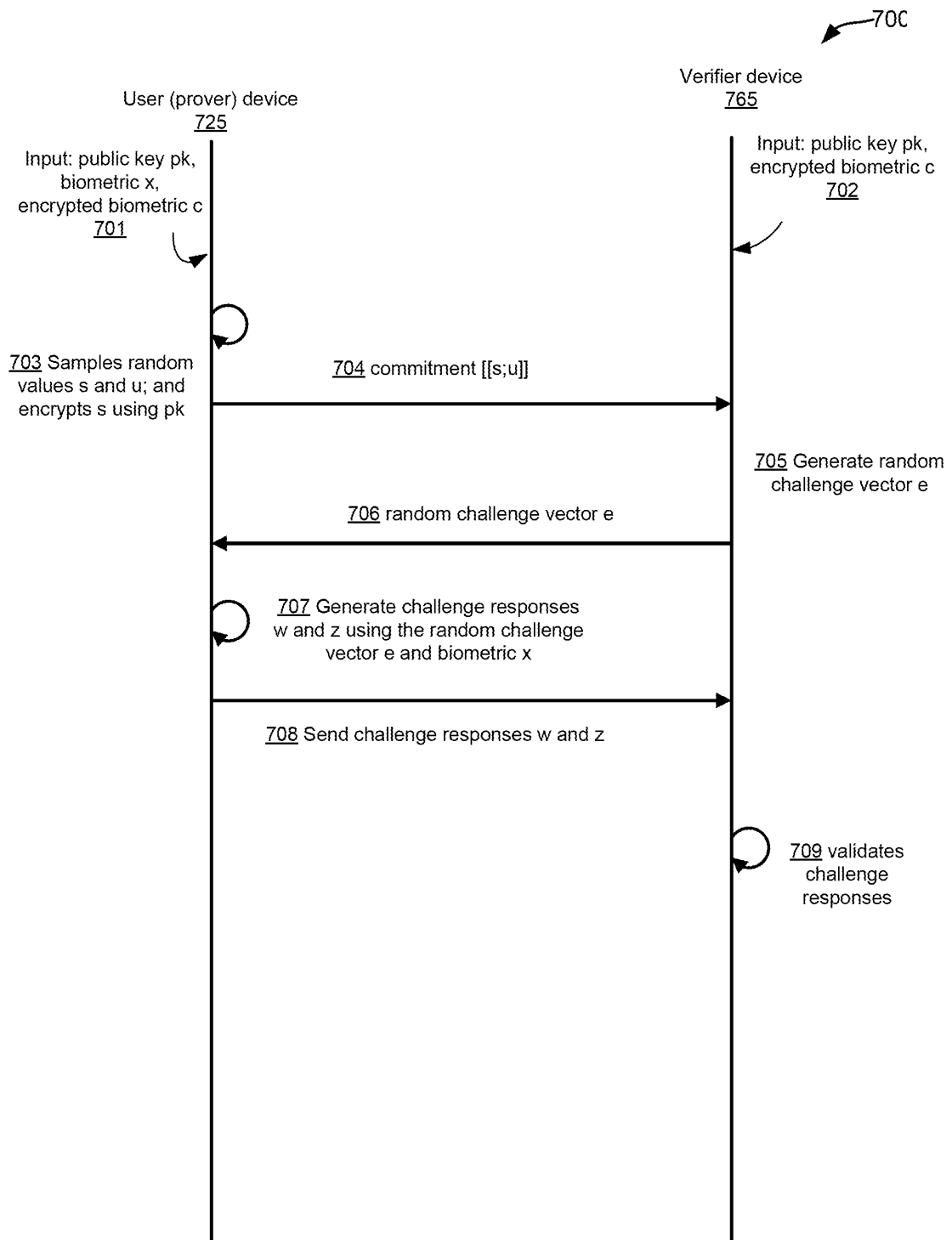


FIG. 7

Parameters: A Paillier public key $pk = (N, g)$, statement $st = (c_1, \dots, c_n)$ and witness $wit = ((x_1, r_1), \dots, (x_n, r_n))$.

Protocol:

810 • P samples $s \leftarrow \mathbb{Z}_N$ and sends $\llbracket s; u \rrbracket \leftarrow \text{Plr.Enc}(pk, s)$ to V .

820 • V sends $e_1, \dots, e_n \leftarrow \mathbb{Z}_N^*$ to P .

830 • P sends $w := s + \sum_i e_i x_i \bmod N$, and $z := ug^t \prod_i r_i^{e_i} \bmod N^2$ to V where t is defined by $tN + w = s + \sum_i e_i x_i$.

840 • V outputs accept if $\llbracket w; z \rrbracket = \llbracket s; u \rrbracket \cdot \prod_i c_i^{e_i}$ and otherwise reject.

FIG. 8

Algorithms:

- $\text{Encode}_1(u_1, \dots, u_n) : \text{Output } u^* = \sum_{i=1}^n 2^{k \cdot (2^{i-1}-1)} u_i \in \mathbb{Z}_{2^k 2^n}.$
- $\text{Add}_1(u^*, v^*) : \text{Output } w^* = (u^* + v^*)$
- $\text{Decode}_1(u^*) : \text{Re-order } u^* \text{ as } u^* = \sum_{i=1}^n 2^{k \cdot (2^{i-1}-1)} u_i, \text{ output } (u_1, \dots, u_n).$
- $\text{Mult}(u^*, v^*) : \text{Output } \tilde{w} = (u^* \cdot v^*)$
- $\text{Encode}_2(u_1, \dots, u_n) : \text{Output } \tilde{u} = \sum_{i=1}^n 2^{k \cdot 2 \cdot (2^{i-1}-1)} u_i \in \mathbb{Z}_{2^k 2^n}.$
- $\text{Add}_2(\tilde{u}, \tilde{v}) : \text{Output } \tilde{w} = (\tilde{u} + \tilde{v})$
- $\text{Mask}(\tilde{w}) \rightarrow \text{Output } \tilde{z} = \left(\tilde{w} + \sum_{j \in I} 2^{j \cdot k} \rho_j \right)$ where $I = \{0, 2, 6, (2^n - 2)\}$ and the ρ_j 's are picked randomly in $\mathbb{Z}_{2^k}.$
- $\text{Decode}_2(\tilde{u}) : \text{Re-order } \tilde{u} = \left(\sum_{i \in I} 2^{i \cdot k} u_i \right) + \left(\sum_{j \in \bar{I}} 2^{j \cdot k} \rho_j \right)$ where $I = \{0, 2, \dots, (2^n - 2)\}.$ Output the n values $\{u_i\}_{i \in I}.$

Here, $k = (m + d + \lambda)$ and $L = (k \cdot 2^n).$

Correctness. Correctness follows easily by inspection.

Security. Given a tuple $(w_1, \dots, w_n)$ computed as $\text{Decode}_2(\text{Mask}(\tilde{w})),$ the simulator outputs a hyper-encoding $\tilde{z}$ as follows:

- Let $I = \{0, 2, \dots, (2^n - 2)\}.$ Set $\{z_i\}_{i \in I} = (w_1, \dots, w_n).$
- $\tilde{z} = \left(\sum_{i \in I} 2^{i \cdot k} z_i \right) + \left(\sum_{j \in \bar{I}} 2^{j \cdot k} \rho_j \right)$ where each ρ_j is picked uniformly at random.

It is easy to observe that $\{\text{Mask}(\tilde{w})\}$ and $\{\text{Sim}(w_1, \dots, w_n)\}$ are identically distributed and hence statistically indistinguishable.

FIG. 9

Enrollment Phase: The device $\mathcal{D}$ has an input vector $\vec{u} = (u_1, \dots, u_\ell)$ such that $\|\vec{u}\| = 1$. The enrollment phase proceeds as follows.

- Device $\mathcal{D}$ does the following:
 - (1) Compute $u_1^*, \dots, u_\ell^*$ as the encoding of $\vec{u}$. That is, $u_i^* = \text{Encode}_1(u_1, \dots, u_n)$, $\dots$, $u_\ell^* = \text{Encode}_1(u_{\ell-n+1}, \dots, u_\ell)$ where $t = \lfloor \frac{\ell}{n} \rfloor$.
 - (2) Compute $(\text{esk}, \text{epk}) \leftarrow \text{Plr.Setup}(1^\lambda)$. For all $i \in [t]$, compute $\llbracket u_i^* \rrbracket = \text{Plr.Enc}(\text{epk}, u_i^*)$.
 - (3) Send $(\text{epk}, \{\llbracket u_i^* \rrbracket\}_{i \in [t]})$ to $\mathcal{SP}$.
- Service provider $\mathcal{SP}$ responds back with a message "Enrolled".

Matching Phase: The terminal $\mathcal{T}$ has an input vector $\vec{w} = (w_1, \dots, w_\ell)$ such that $\|\vec{w}\| = 1$. The matching phase proceeds as follows.

- **Round 1:** ($\mathcal{D} \rightarrow \mathcal{T}$) Device $\mathcal{D}$ sends $(\text{epk}, \{\llbracket u_i^* \rrbracket\}_{i \in [t]})$ to $\mathcal{T}$.
- **Round 2:** ($\mathcal{T} \rightarrow \mathcal{D}$) $\mathcal{T}$ does the following:
 - (1) Compute $(w_1^*, \dots, w_\ell^*) = \text{Encode}_1(\vec{w})$.
 - (2) Compute $\llbracket \vec{z} \rrbracket = \llbracket \sum_{i=1}^t (u_i^* \cdot w_i^*) \rrbracket$ using the algorithm $\text{Plr.Add}(\cdot)$. Note that $\vec{z}$ is now a level-2 encoding.
 - (3) Pick n random k -bit strings $\rho_1, \dots, \rho_n$ such that $(\rho_1 + \dots + \rho_n) = 0$. Let $\tilde{\rho} = \text{Encode}_2(\rho_1, \dots, \rho_n)$.
 - (4) Compute and send $\llbracket \tilde{\rho} \rrbracket = \llbracket \text{Mask}(\vec{z} + \tilde{\rho}) \rrbracket$ to $\mathcal{D}$.
- **Round 3:** ($\mathcal{D} \rightarrow \mathcal{T}$) $\mathcal{D}$ does the following:
 - (1) Compute $(\text{IP}_1, \dots, \text{IP}_n) = \text{Decode}_2(\text{Plr.Dec}(\text{esk}, \llbracket \tilde{\rho} \rrbracket))$. Set $\text{IP} = \sum_{i=1}^n \text{IP}_i$.
 - (2) If $\text{IP} \geq T$, $\mathcal{D}$ sends 1 to $\mathcal{T}$. Else, sends 0.
- **Output Computation.** Both parties output the bit sent in the last round.

FIG. 10

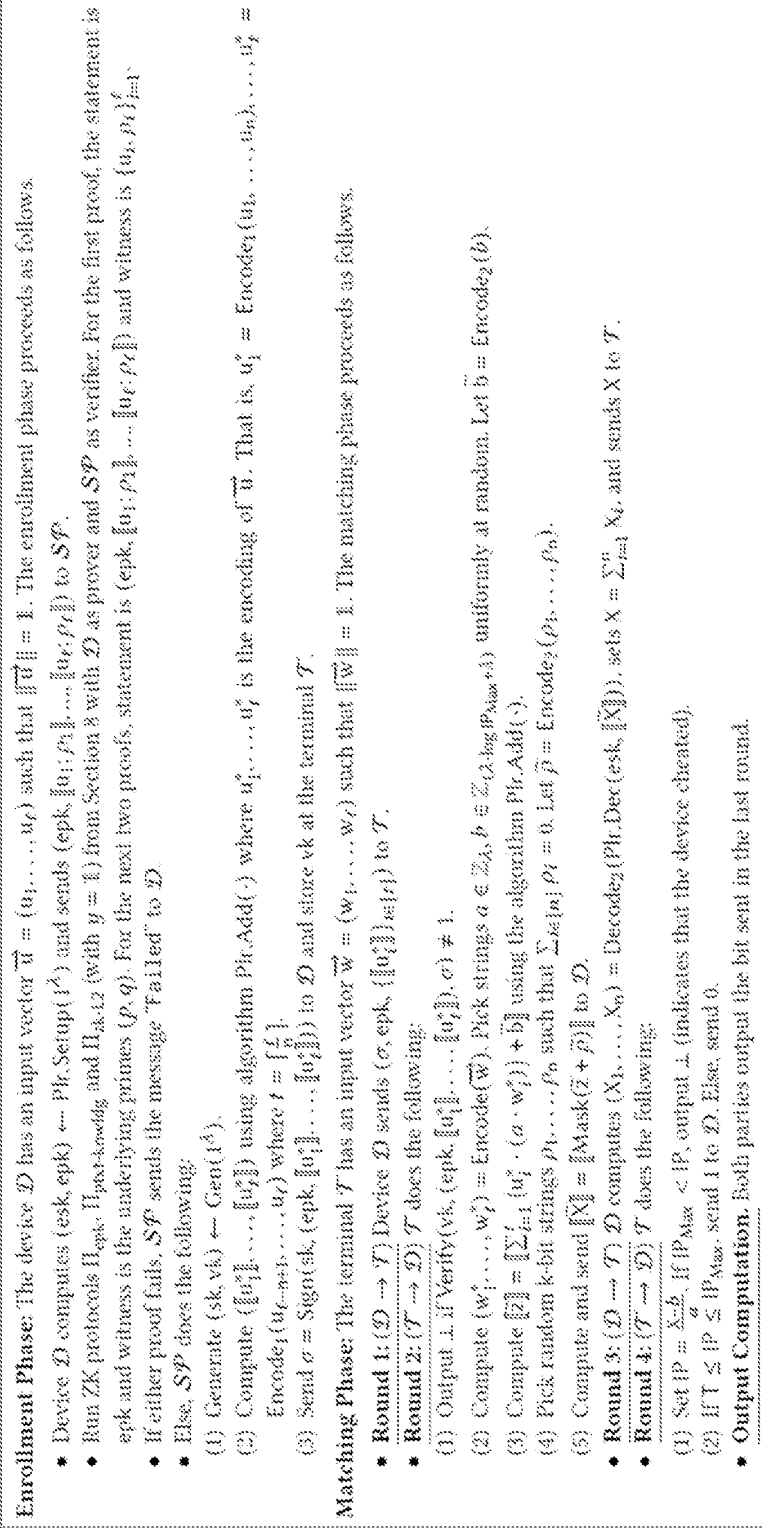


FIG. 11

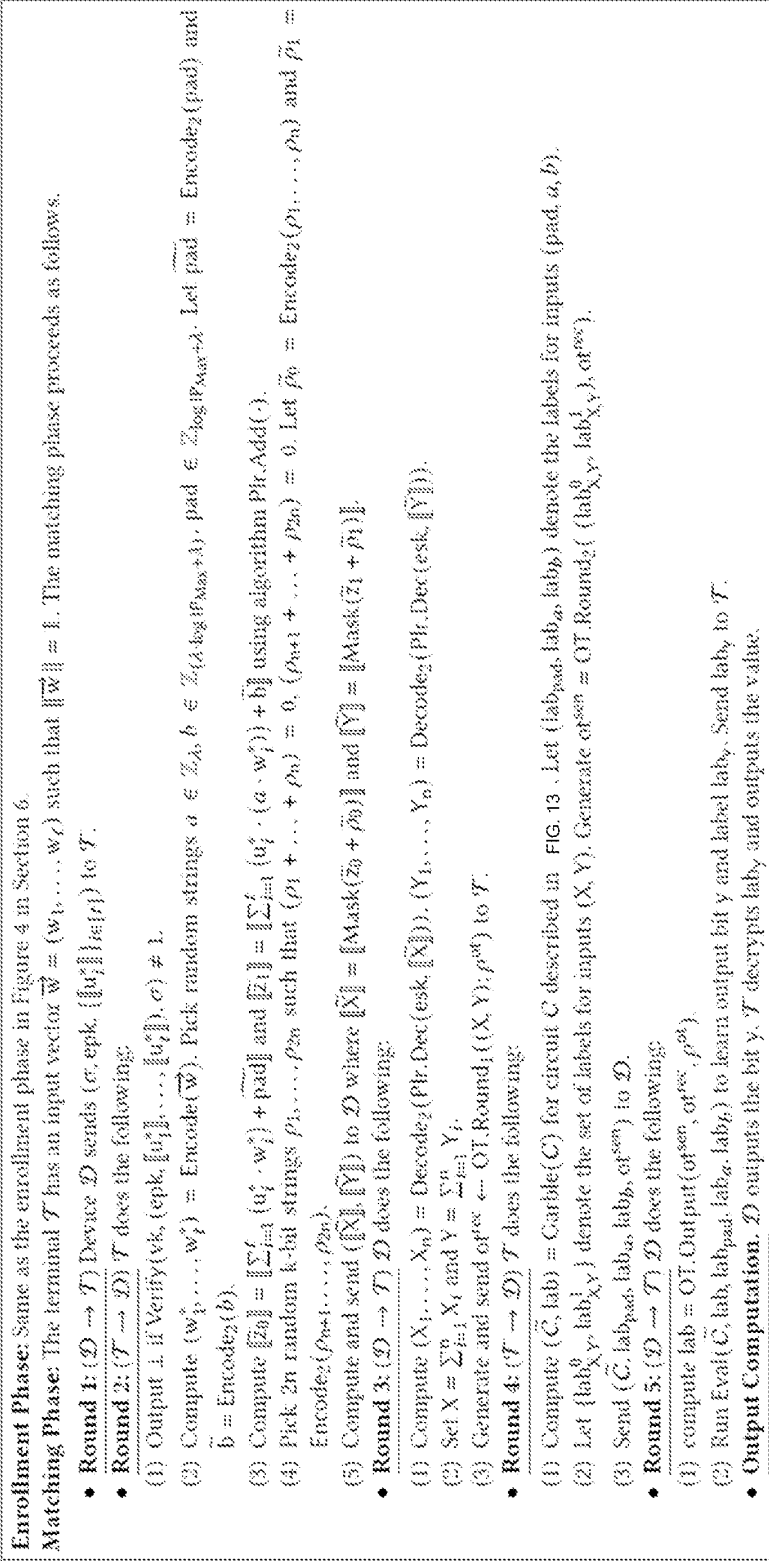


FIG. 12

13/16

Inputs: (X, Y, pad, a, b) .

Computation:

- Compute $IP = (X - \text{pad})$. If $(a \cdot IP + b) \neq Y$, output $\perp$.
- If $T \leq IP$, output 1. Else, output 0.

FIG. 13

m	ℓ	Π_1		Π_2			Π_3			Π_{en}		
		m_1	$T \rightarrow \mathcal{D}$	m_1	$\mathcal{D} \rightarrow T$	$T \rightarrow \mathcal{D}$	m_1	$\mathcal{D} \rightarrow T$	$T \rightarrow \mathcal{D}$	$\mathcal{D} \rightarrow S\mathcal{P}$	$S\mathcal{P} \rightarrow \mathcal{D}$	
8	128	8,464	516	11,536	260	516	11,536	2,944	66,809	124,007	66,082	
	256	16,656	516	22,288	260	516	22,288	2,944	69,561	189,543	131,618	
	512	33,040	516	44,304	260	516	44,304	3,071	72,313	320,615	262,690	
	1024	65,808	516	87,824	260	516	87,824	3,076	75,065	582,760	524,834	
16	128	8,464	516	13,584	260	516	13,584	4,000	110,841	124,007	66,082	
	256	16,656	516	26,896	260	516	26,896	4,000	113,593	189,543	131,618	
	512	33,040	516	53,008	260	516	53,008	4,132	116,313	320,615	262,690	
	1024	65,808	516	105,232	260	516	105,232	4,132	119,033	582,760	524,834	
24	128	10,000	516	16,656	260	516	16,656	5,056	154,393	124,007	66,082	
	256	19,216	516	33,040	260	516	33,040	5,056	157,113	189,543	131,618	
	512	38,160	516	65,808	260	516	65,808	5,188	159,833	320,615	262,690	
	1024	75,536	516	131,344	260	516	131,344	5,188	162,553	582,760	524,834	

FIG. 14

15/16

m	ℓ	Π_1 -Enc	Π_1	Π_{en} -Enc	Π_{en}	Π_2	Π_3
8	128	318	12	1,110	2,914	25	71
	256	626	19	2,198	5,018	44	102
	512	1,118	40	4,389	9,249	74	171
	1,024	2,263	72	8,809	17,704	138	289
16	128	457	15	1,082	2,927	33	87
	256	630	24	2,200	5,008	49	119
	512	1,131	45	4,367	9,439	89	195
	1,024	2,316	88	8,855	17,849	170	357
24	128	384	19	1,094	2,892	28	74
	256	742	32	2,204	5,053	43	105
	512	1,298	64	4,373	9,242	73	164
	1,024	2,541	118	8,772	17,822	123	266

FIG. 15

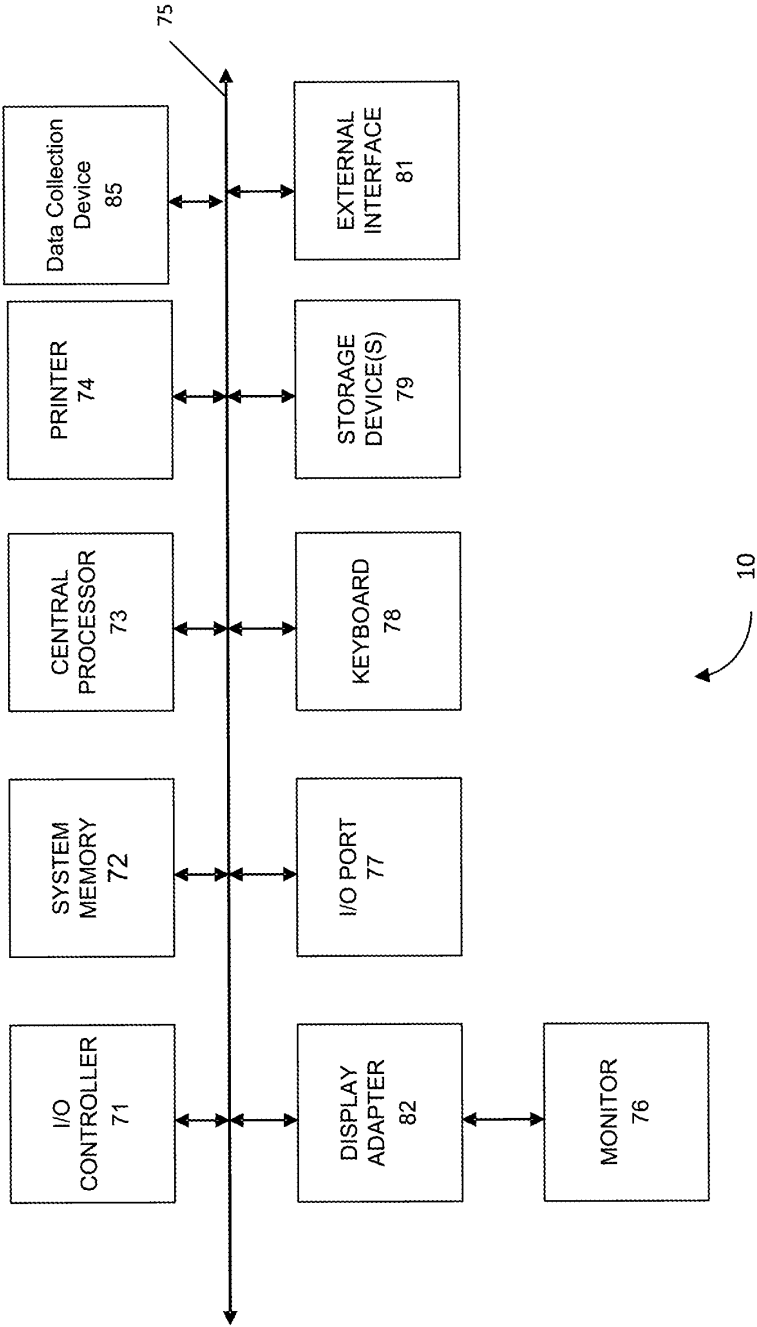


FIG. 16