디지털 컨텐츠 거래 시스템

Privacy preserved digital contents trading system on public blockchain

1. Contribution

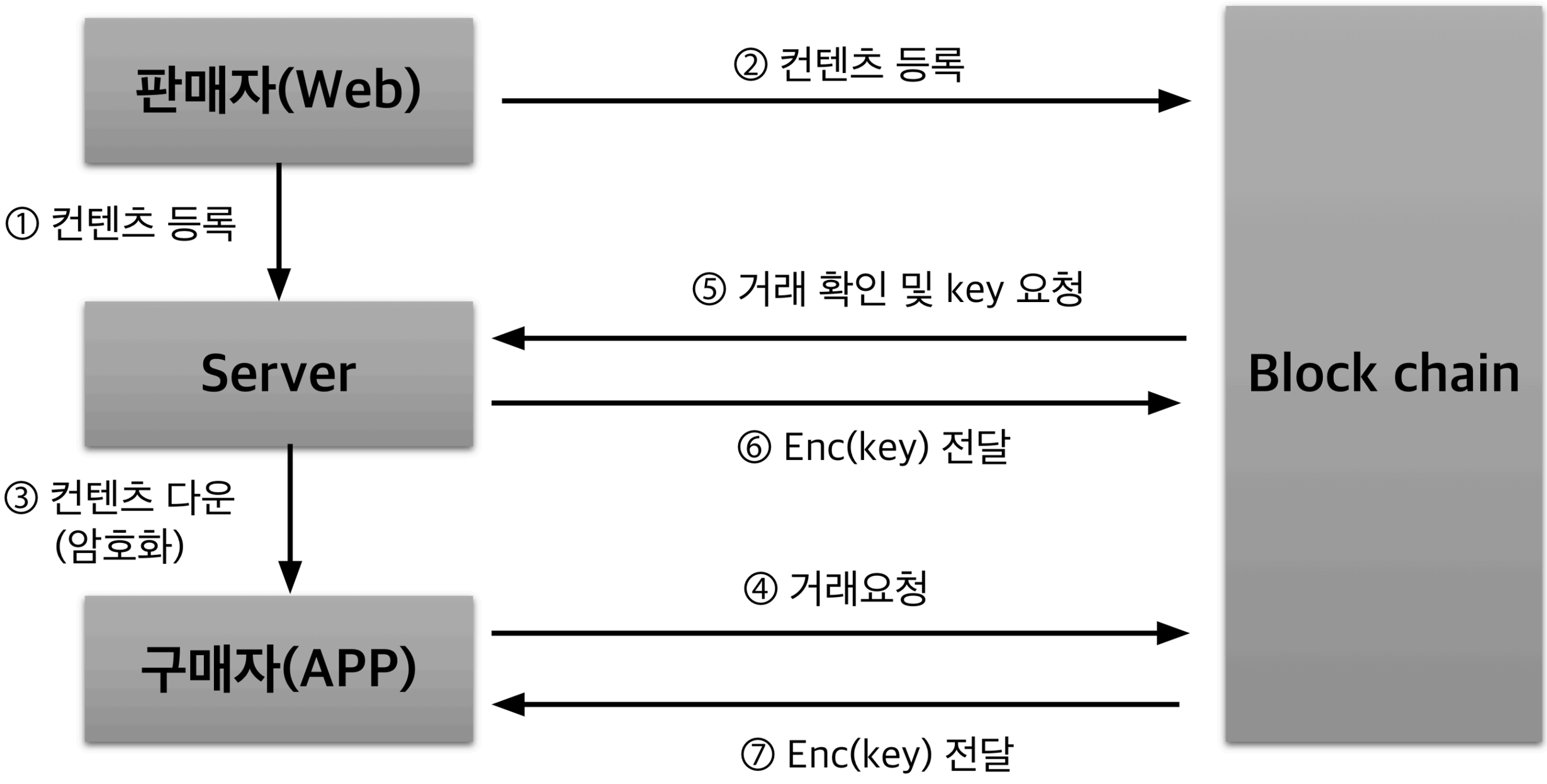
(1) 거래내역의 투명성

- 디지털 컨텐츠를 대리 판매하는 시스템이 거래량을 조작할 수 없음

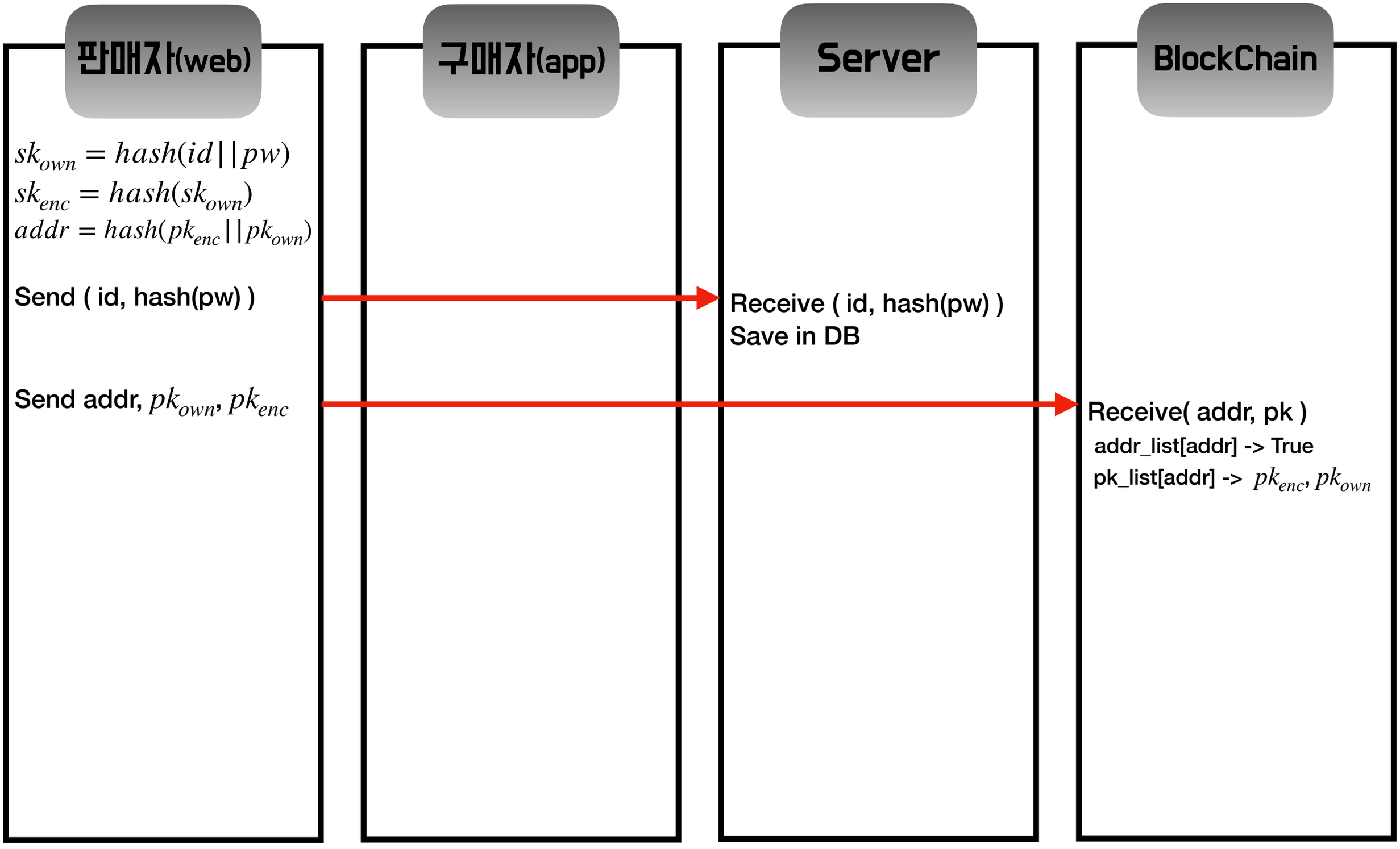
(2) 프라이버시 거래

- 구매목록, 구매한 컨텐츠 정보 등 거래 대한 프라이버시를 보호

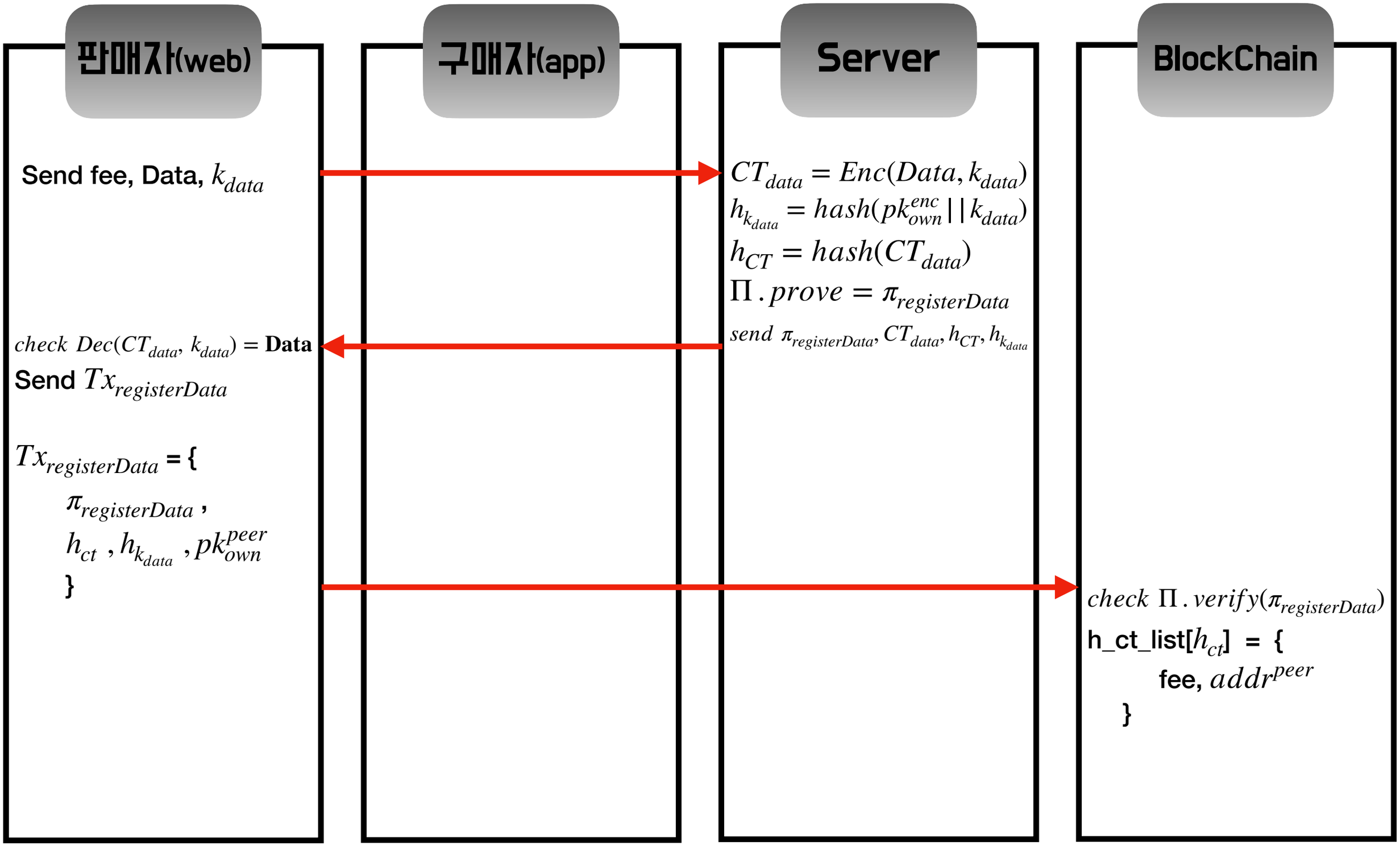
2. Component


판매자(Web)
② 컨텐츠 등록
① 컨텐츠 등록
⑤ 거래 확인 및 key 요청
Server
Block chain
⑥ Enc(key) 전달
③ 컨텐츠 다운
(암호화)
④ 거래요청
구매자(APP)
⑦ Enc(key) 전달

3-1 Register User



3-2 Register Data



3-2 Register Data

판매자(web)

구매자(app)

Server

BlockChain

Send fee, Data, k_{data}

$CT_{data} = Enc(Data, k_{data})$

$h_{k_{data}} = hash(pk_{own}^{peer} || k_{data})$

$h_{CT} = hash(CT_{data})$

$\Pi.prove = \pi_{registerData}$

$send\ \pi_{registerData}, CT_{data}, h_{CT}, h_{k_{data}}$

$check\ Dec(CT_{data}, k_{data}) = \textbf{Data}$

Send $Tx_{registerData}$

$Tx_{registerData}$ = {

$\pi_{registerData}$,

h_{ct} , $h_{k_{data}}$, pk_{own}^{peer}

}

Relation$_{RegisterData}$

Inputs :

h_{ct} : hashed cipher Text

$h_{k_{data}}$: hashed k_{data}

pk_{own}^{peer} : data owner's pk_own

Witness:

data : text Data

CT_{data} :encrypted Data

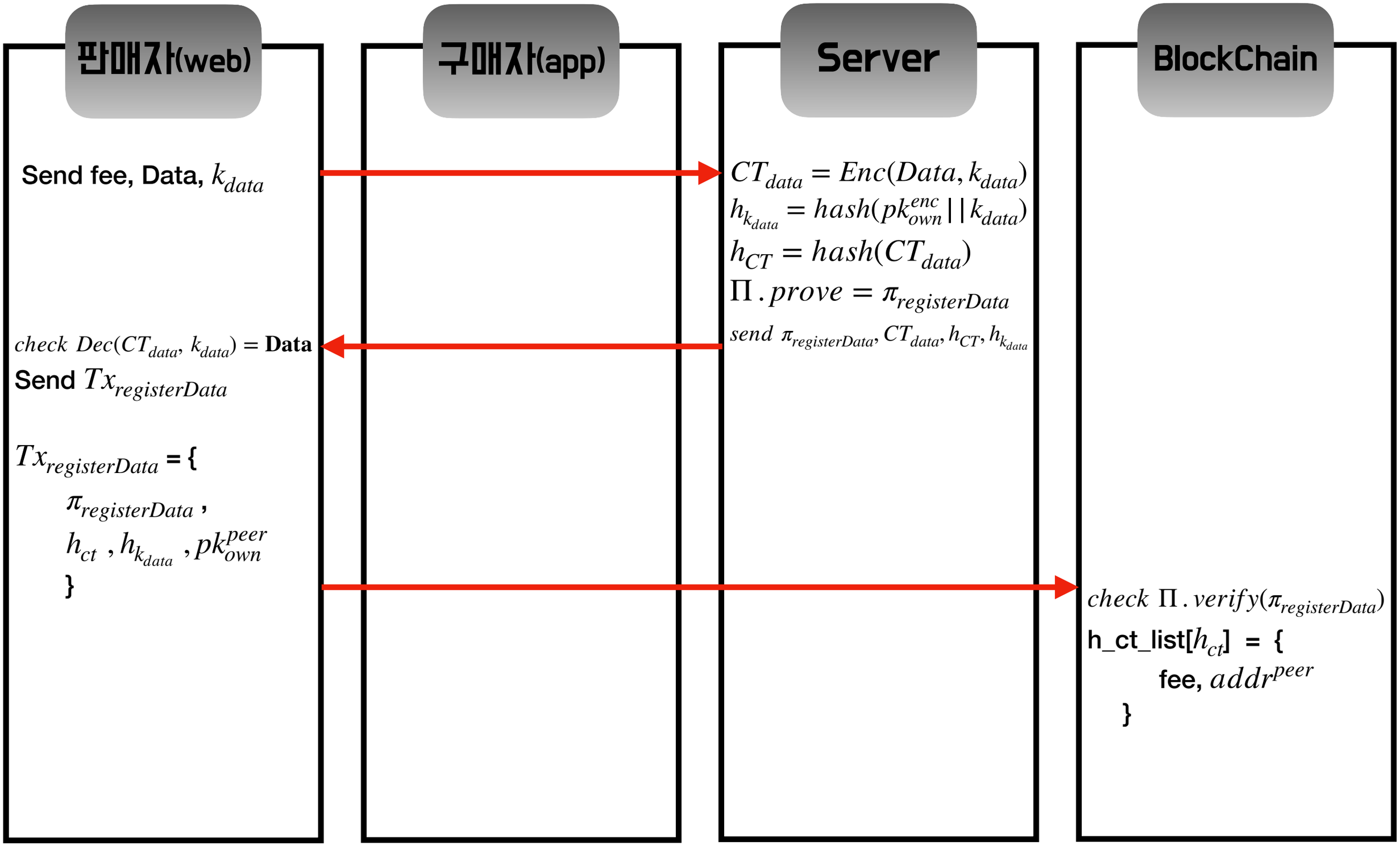
k_{data} : key used in symmetric-key encryption scheme

$h_{kdata} = \text{Hash}(\ pk_{own}^{peer} \parallel k_{data}\ )$

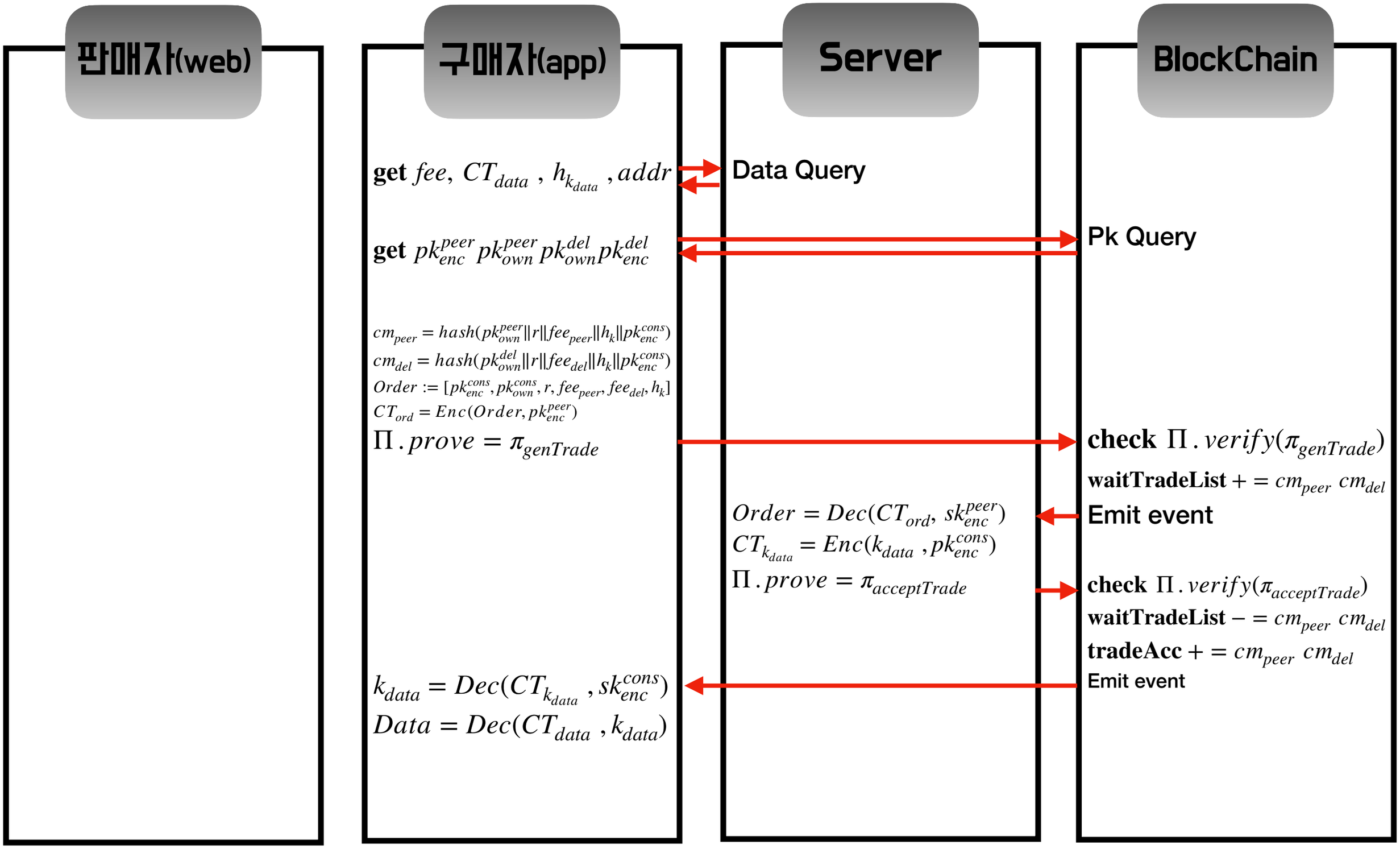
$h_{ct} = \text{Hash}(\ CT_{data}\ )$

$CT_{data} = \text{SE.Enc}(\text{data}, k_{data})$

3-2 Register Data



3-3 Trade Data



3-3 Trade Data

판매자(web)

구매자(app)

get fee, CT_{data} , $h_{k_{data}}$

get pk_{enc}^{peer} pk_{own}^{peer} pk_{own}^{del} pk_{enc}^{del}

$cm_{peer} = hash(pk_{own}^{peer} \| r \| fee_{peer} \| h_k \| pk_{enc}^{cons})$
$cm_{del} = hash(pk_{own}^{del} \| r \| fee_{del} \| h_k \| pk_{enc}^{cons})$
$Order := [pk_{enc}^{cons}, pk_{own}^{cons}, r, fee_{peer}, fee_{del}, h_k]$
$CT_{ord} = Enc(Order, pk_{enc}^{peer})$
$\Pi.prove = \pi_{genTrade}$

$k_{data} = Dec(CT_{k_{data}}, sk_{enc}^{cons})$
$Data = Dec(CT_{data}, k_{data})$

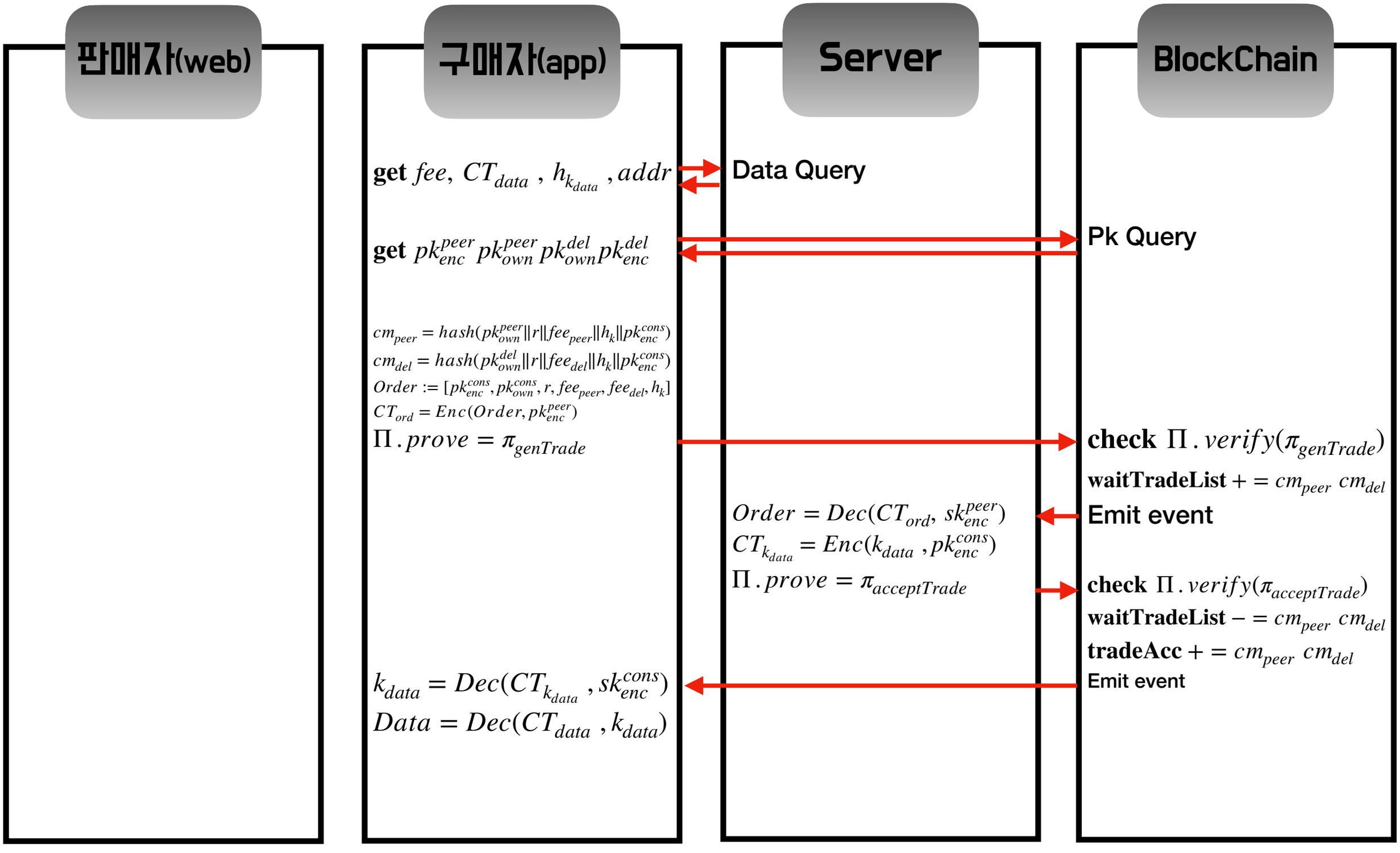
$\text{Relation}_{genTrade}$

Inputs :
- cm_{peer} : coin commitment send to data owner(peer)
- cm_{del} : coin commitment send to delegate server
- CT_{ord} : encrypted buyer's key
- ENA : encrypted account value before buy
- ENA′ : encrypted account value after buy
- fee_{del} : delegation fee
- fee_{peer} : data fee

Witness:
- r : random value to make cm
- h_k : hashed k_{data}
- pk_{own}^{del}: to make cm_{del}
- pk_{enc}^{cons}: consumer's pk_{enc} to make CT_{keys}
- pk_{own}^{cons} : consumer's pk_{own} to make CT_{keys}
- pk_{enc}^{peer}: key used in public-key encryption
- pk_{own}^{peer} : to make cm_{peer}
- k_{ENA}: key used in symmetric-key encryption scheme

$fee_{del} + fee_{peer} = \text{SE.Dec(ENA, } k_{ENA}) - \text{SE.Dec(ENA′,k}_{ENA})$
$cm_{peer} = \text{Hash}(pk_{own}^{peer} \| r \| fee_{peer} \| h_k \| pk_{enc}^{cons})$
$cm_{del} = \text{Hash}(pk_{own}^{del} \| r \| fee_{del} \| h_k \| pk_{enc}^{cons})$
$Order := [\, pk_{enc}^{cons}, pk_{own}^{cons}, r, fee_{peer}, fee_{del}, h_k \,]$
$CT_{ord} = \text{PE.Enc}(\, Order, pk_{enc}^{peer} \,)$

3-3 Trade Data



3-3 Trade Data

판매자(web) | 구매자(app) | Server | BlockChain

Data Query

Pk Query

$\underline{\text{Relation}_{AcceptTrade}}$

Inputs :

cm_{peer} : *coin commitment*

cm_{own} : *coin commitment*

$CT_{k_{data}}$: To inform the consumer of the k_{data}

Witness:

h_k : *hashed* k_{data}

k_{data} : key that make CT_{data}

pk_{onw}^{peer} : data owner's pk_{own}

pk_{enc}^{cons} : to encrypt k_{data} consumer's pk_{enc}

r : to prove coin commitment ownership

fee_{del} : to prove coin commitment ownership

fee_{peer} : to prove coin commitment ownership

$cm_{peer} = \text{Hash}(pk_{own}^{peer} \| r \| fee_{peer} \| h_k \| pk_{enc}^{cons})$

$cm_{del} = \text{Hash}(pk_{own}^{del} \| r \| fee_{del} \| h_k \| pk_{enc}^{cons})$

$h_k = \text{Hash}(\ pk_{own}^{peer} \ \|\ k_{data}\ )$

$CT_{k_{data}} = \text{PE.Enc}(k_{data}, pk_{enc}^{cons})$

check $\Pi.verify(\pi_{genTrade})$

waitTradeList $+= cm_{peer}\ cm_{del}$

Emit event

$Order = Dec(CT_{ord}, sk_{enc}^{peer})$

$CT_{k_{data}} = Enc(k_{data}, pk_{enc}^{cons})$

$\Pi.prove = \pi_{acceptTrade}$

check $\Pi.verify(\pi_{acceptTrade})$

waitTradeList $-= cm_{peer}\ cm_{del}$

tradeAcc $+= cm_{peer}\ cm_{del}$

Emit event

3-3 Trade Data