qpigeon Protocol

Definitions

$$
\begin{aligned}
PS &: \text{Public key (for signing)} \\
SS &: \text{Secret key (for signing)} \\
PK &: \text{Public key (for KEM)} \\
SK &: \text{Secret key (for KEM)} \\
K &: \text{Symmetric encryption key} \\
N &: \text{List of used nonces}
\end{aligned}
$$

$\text{Sign}_{SS}(M) = S$	Signs message M using private key SS creating signature S.
$\text{Sign}^{-1}_{PS}(S, M) = \{0, 1\}$	Verifies message M matches signature S using public key PS. Outputs 1 when signature matches.
$\text{KEM}_{PK}(K) = C$	Encrypts the given key K using public key PK.
$\text{KEM}^{-1}_{SK}(C) = K$	Decrypts the given encrypted key C using secret key SK. KEM stands for Key Encapsulation Mechanism.
$\text{Enc}_K(M) = C$	Encrypts the given message using symmetric key K.
$\text{Enc}^{-1}_K(C) = M$	Decrypts the given ciphertext using symmetric key K.
$\text{Now}() = T$	Outputs the current timestamp.

Contact Request and Accept

Bob adds Alice as a contact.

$$\begin{aligned}
\text{Bob has} &: SS_{\text{Bob}}, PS_{\text{Alice}}, T_{\text{Threshold}}, N \\
\text{Server has} &: PS_{\text{Bob}}, PS_{\text{Alice}}, T_{\text{Threshold}}, N \\
\text{Alice has} &: SS_{\text{Alice}}, PS_{\text{Bob}}, T_{\text{Threshold}}, N
\end{aligned}$$

Bob calculates :

$T = \text{Now}()$	Get current timestamp.
$n = \{0,1\}^{128} \text{ s.t. } (n, PS_{\text{Bob}}) \notin N$	Generate nonce.
$N = N \cup \{(n, PS_{\text{Bob}})\}$	Add nonce to list.
$S = \text{Sign}_{SS_{\text{Bob}}}(T \vert\vert n \vert\vert PS_{\text{Alice}})$	Sign contact request.

Bob sends to server : $S, T, n, PS_{\text{Alice}}$

Server calculates :

$S_{\text{Verify}} = \text{Sign}^{-1}_{PS_{\text{Bob}}}(S, T \vert\vert n \vert\vert PS_{\text{Alice}})$	Verify contact request is from Bob.
$T > \text{Now}() - T_{\text{Threshold}}$	Verify contact request is recent.
$(n, PS_{\text{Bob}}) \notin N$	Verify nonce is new.
$N = N \cup \{(n, PS_{\text{Bob}})\}$	Add old nonce to list.

Server sends to Alice : S, T, n

Alice calculates :

$S_{\text{Verify}} = \text{Sign}^{-1}_{PS_{\text{Bob}}}(S, T \vert\vert n \vert\vert PS_{\text{Alice}})$	Verify contact request is from Bob. If $S_{\text{Verify}} = 0$, reject.
$T > \text{Now}() - T_{\text{Threshold}}$	Verify contact request is recent.
$(n, PS_{\text{Bob}}) \notin N$	Verify nonce is new.
$N = N \cup \{(n, PS_{\text{Bob}})\}$	Add old nonce to list.
$T = \text{Now}()$	Get current timestamp.
$n = \{0,1\}^{128} \text{ s.t. } (n, PS_{\text{Alice}}) \notin N$	Generate nonce.
$N = N \cup \{(n, PS_{\text{Alice}})\}$	Add nonce to list.
$S = \text{Sign}_{SS_{\text{Alice}}}(T \vert\vert n \vert\vert PS_{\text{Bob}})$	Sign contact request.

Alice sends to server : S, T, n, PS_{Bob}

Server calculates :

$$
\begin{aligned}
S_{\text{Verify}} &= \text{Sign}^{-1}_{PS_{\text{Alice}}}(S, T||n||PS_{\text{Bob}}) && \text{Verify contact request is from Alice.} \\
& && \text{If } S_{\text{Verify}} = 0\text{, reject.} \\
T &> \text{Now}() - T_{\text{Threshold}} && \text{Verify contact request is recent.} \\
(n, PS_{\text{Alice}}) &\notin N && \text{Verify nonce is new.} \\
N &= N \cup \{(n, PS_{\text{Alice}})\} && \text{Add old nonce to list.}
\end{aligned}
$$

Server sends to Bob : S, T, n

Bob calculates :

$$
\begin{aligned}
S_{\text{Verify}} &= \text{Sign}^{-1}_{PS_{\text{Alice}}}(S, T||n||PS_{\text{Bob}}) && \text{Verify contact request is from Alice.} \\
& && \text{If } S_{\text{Verify}} = 0\text{, reject.} \\
T &> \text{Now}() - T_{\text{Threshold}} && \text{Verify contact request is recent.} \\
(n, PS_{\text{Alice}}) &\notin N && \text{Verify nonce is new.} \\
N &= N \cup \{(n, PS_{\text{Alice}})\} && \text{Add old nonce to list.}
\end{aligned}
$$

Public Key (for KEM) Distribution

Alice sends a public key (for KEM) PK_{Alice} to Bob.

Alice has : $SS_{\text{Alice}}, PK_{\text{Alice}}$
Server has : PS_{Alice}
Bob has : PS_{Alice}

Alice calculates :

$$S = \text{Sign}_{SS_{\text{Alice}}}(PK_{\text{Alice}})$$ Signs public key.

Alice sends to Server : S, PK_{Alice}

Server calculates :

$$S_{\text{Verify}} = \text{Sign}^{-1}_{PS_{\text{Alice}}}(S, PK_{\text{Alice}})$$ Verify message is from Alice.
If $S_{\text{Verify}} = 0$, reject.

Server sends to Bob : S, PK_{Alice}

Bob calculates:

$$S_{\text{Verify}} = \text{Sign}^{-1}_{PS_{\text{Alice}}}(S, PK_{\text{Alice}})$$ Verify message is from Alice.
If $S_{\text{Verify}} = 0$, reject.

Bob sends message to Alice

Bob sends a given message M to Alice.

$$\begin{aligned}\text{Bob has} &: SS_{\text{Bob}}, PK_{\text{Alice}}, N \\ \text{Server has} &: PS_{\text{Bob}}, T_{\text{Threshold}}, N \\ \text{Alice has} &: SK_{\text{Alice}}, PS_{\text{Bob}}, T_{\text{Threshold}}, N\end{aligned}$$

Bob calculates :

$K = \{0,1\}^n$	Generates key of length n.
$C_K = \text{KEM}_{PK_{\text{Alice}}}(K)$	Encrypts key.
$C_M = \text{Enc}_K(M)$	Encrypts message.
$T = \text{Now}()$	Get current timestamp.
$n = \{0,1\}^{128}$ s.t. $(n, PS_{\text{Bob}}) \notin N$	Generate nonce.
$N = N \cup \{(n, PS_{\text{Bob}})\}$	Add nonce to list.
$S = \text{Sign}_{SS_{\text{Bob}}}(T\|\|n\|\|C_K\|\|C_M)$	Sign message.

Bob sends to server : S, T, n, C_K, C_M

Server calculates :

$S_{\text{Verify}} = \text{Sign}^{-1}_{PS_{\text{Bob}}}(S, T\|\|n\|\|C_K\|\|C_M)$	Verify message is from Bob.
	If $S_{\text{Verify}} = 0$, reject.
$T > \text{Now}() - T_{\text{Threshold}}$	Verify message is recent.
$(n, PS_{\text{Bob}}) \notin N$	Verify nonce is new.
$N = N \cup \{(n, PS_{\text{Bob}})\}$	Add old nonce to list.

Server sends to Alice : S, T, n, C_K, C_M

Alice calculates :

$S_{\text{Verify}} = \text{Sign}^{-1}_{PS_{\text{Bob}}}(S, T\|\|n\|\|C_K\|\|C_M)$	Verify message is from Bob.
	If $S_{\text{Verify}} = 0$, reject.
$T > \text{Now}() - T_{\text{Threshold}}$	Verify message is recent.
$(n, PS_{\text{Bob}}) \notin N$	Verify nonce is new.
$N = N \cup \{(n, PS_{\text{Bob}})\}$	Add old nonce to list.
$K = \text{KEM}^{-1}_{SK_{\text{Alice}}}(C_K)$	Decrypt key.
$M = \text{Enc}^{-1}_K(C_M)$	Decrypt message.