Provable Secure Constant-Round Group Key Agreement Protocol based on Secret Sharing

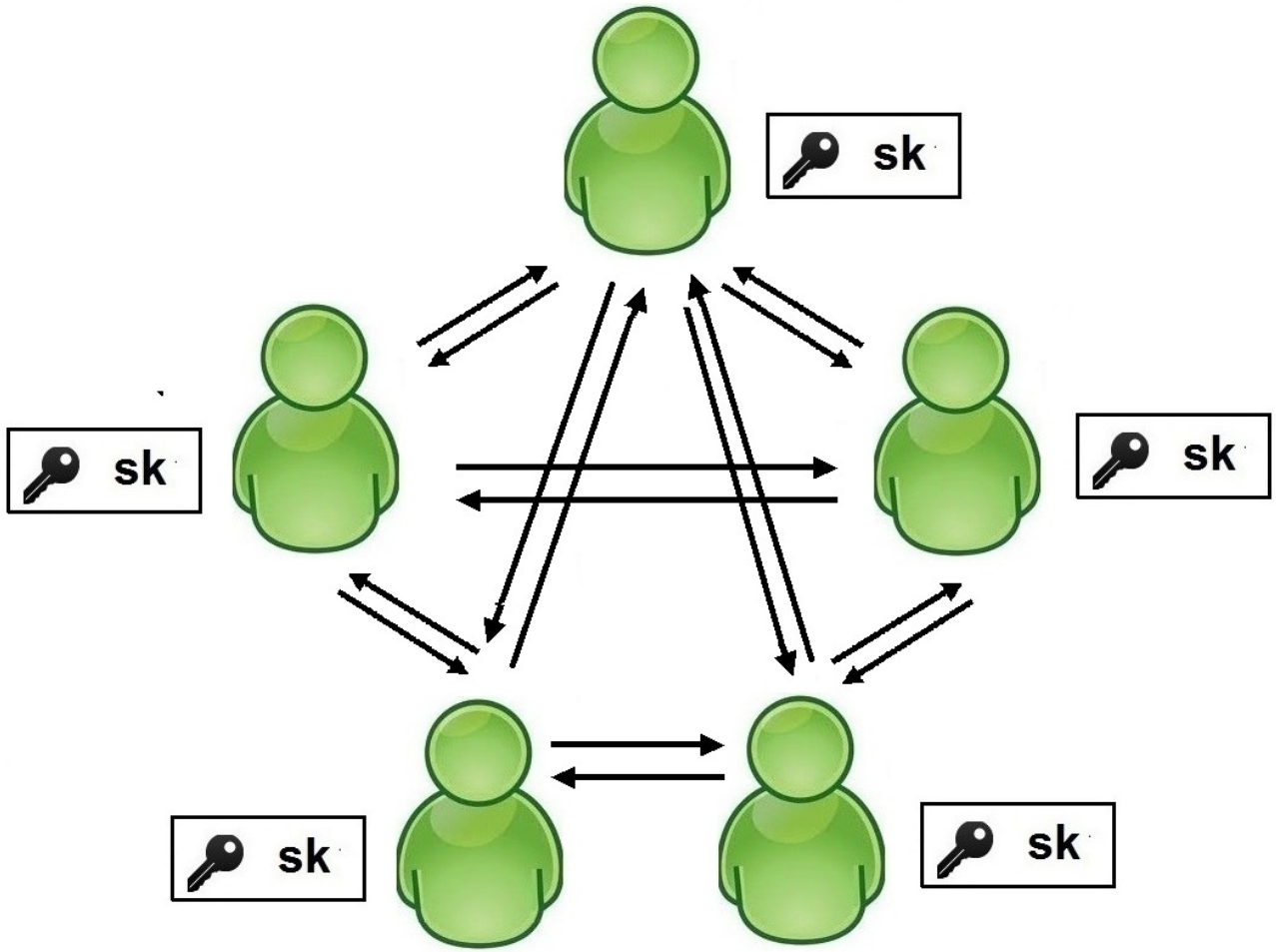
Ruxandra F. Olimid

University of Bucharest
ruxandra.olimid@fmi.unibuc.ro

CISIS 2013, 11 September

GKA (Group Key Agreement)

Goal: Allow multiple users to share a common secret key for subsequent cryptographic use.



Our work

- We define a GKA protocol based on secret n-sharing

Our work

- We define a GKA protocol based on secret n-sharing
- We prove its security in the GBG model

Our work

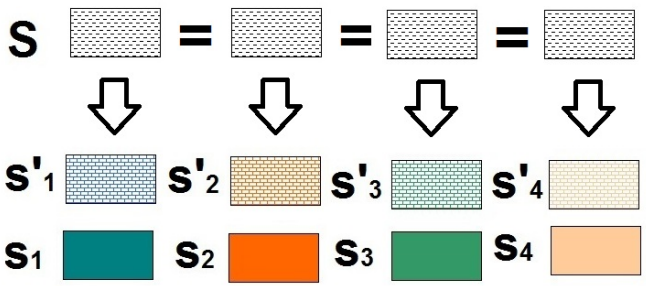
- We define a GKA protocol based on secret n-sharing
- We prove its security in the GBG model
- We analyze its efficiency and compare it to existing work

Our work

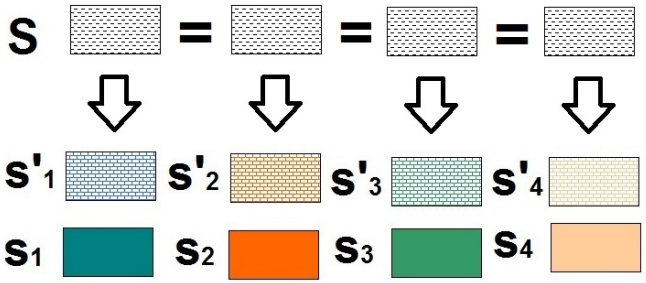
- We define a GKA protocol based on secret n-sharing
- We prove its security in the GBG model
- We analyze its efficiency and compare it to existing work

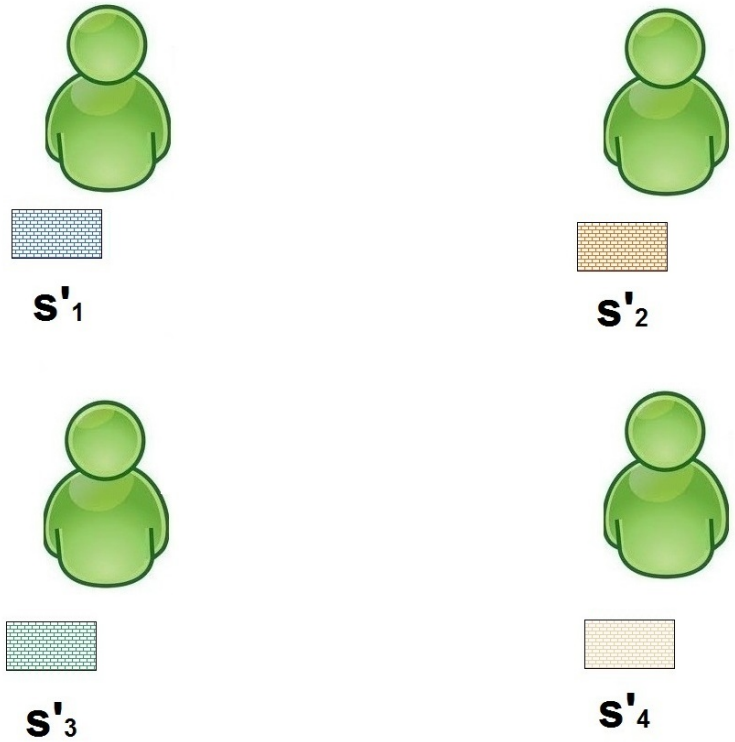
Motivation: few proved secure GKA protocols based on secret sharing exist

Secret n-Sharing

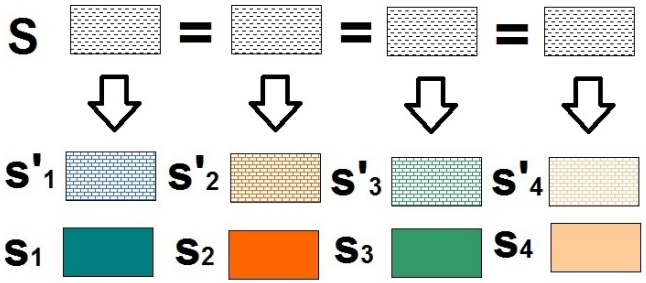


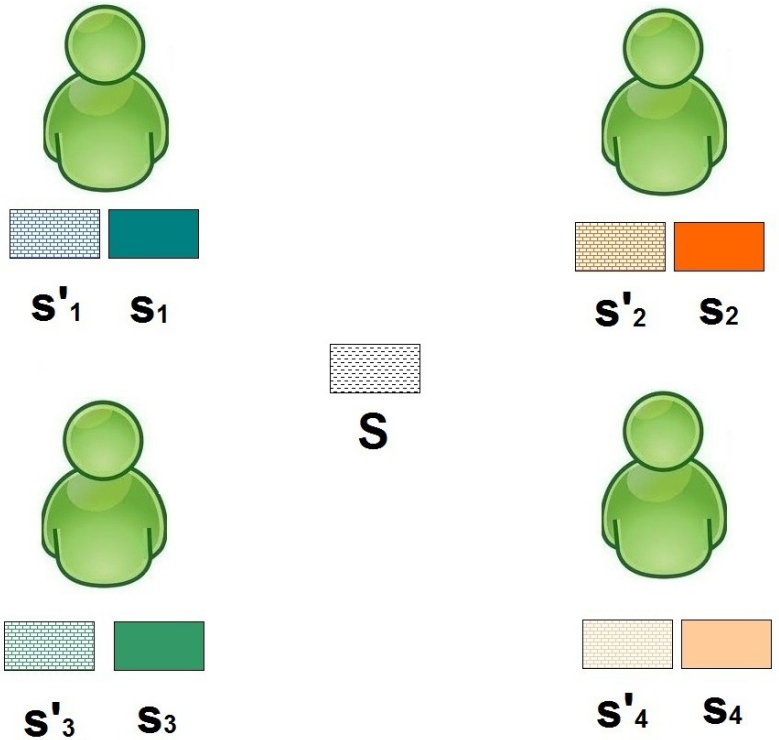
Secret n-Sharing





Secret *n*-Sharing





Notations

- The set of users: $\mathcal{U} = \{U_1, \ldots, U_n\}$
- The initiator: U_1
- U_i's signature: $\Sigma = (\mathsf{Sign}_{U_i}, \mathsf{Verify}_{U_i})$. $i = 1 \ldots n$
- A family of trapdoor functions: $\mathcal{F} = (\mathsf{Gen}, \mathsf{F}(\mathsf{pk}, \cdot), \mathsf{F}^{-1}(\mathsf{sk}, \cdot))$
- A secret n-sharing scheme: $\mathcal{SS} = (\mathsf{Share}, \mathsf{Rec})$

Olimid GKA

1. **Round 1.** U_1:
   runs $\mathcal{F}$.Gen to obtain (pk, sk), $r_1 \leftarrow^R \{0,1\}^{\mathcal{K}}$
   $U_1 \rightarrow^*$: $(\mathcal{U}, pk, \mathsf{F}(pk, r_1), \sigma = \Sigma.\mathsf{Sign}_{U_1}(\mathcal{U}||pk||\mathsf{F}(pk, r_1)))$
2. **Round 2.** U_i, $i = 2, \ldots, n$:
   checks that $\Sigma.\mathsf{Verify}_{U_1}(\mathcal{U}||pk||\mathsf{F}(pk, r_1), \sigma) = 1$, $r_i \leftarrow^R \{0,1\}^{\mathcal{K}}$
   $U_i \rightarrow^*$: $(\mathsf{F}(pk, r_i), \sigma_i = \Sigma.\mathsf{Sign}_{U_i}(\mathcal{U}||pk||\mathsf{F}(pk, r_i)))$
3. **Round 3.** U_1:
   checks that $\Sigma.\mathsf{Verify}_{U_i}(\mathcal{U}||pk||\mathsf{F}(pk, r_i), \sigma_i) = 1$, $i = 2, \ldots, n$
   computes $\mathsf{sid}_{U_1} = \mathsf{F}(pk, r_1)|| \ldots ||\mathsf{F}(pk, r_n)$, $r_i = \mathsf{F}^{-1}(sk, \mathsf{F}(pk, r_i))$
   $k = \mathsf{H}(\mathsf{sid}_{U_1}||r_1||r_2|| \ldots ||r_n)$, r_i' s.t. $\mathcal{SS}.\mathsf{Share}(k, 2) = \{r_i, r_i'\}$
   $U_1 \rightarrow^*$: $(r_2', \ldots r_n', \sigma' = \Sigma.\mathsf{Sign}_{U_1}(\mathcal{U}||pk||r_2'|| \ldots ||r_n'||\mathsf{sid}_{U_1}))$
4. **Key Computation.** U_i, $i = 2, \ldots, n$:
   checks that $\Sigma.\mathsf{Verify}_{U_j}(\mathcal{U}||pk||\mathsf{F}(pk, r_j), \sigma_j) = 1$, $j = 2, \ldots n$, $j \neq i$
   computes $\mathsf{sid}_{U_i} = \mathsf{F}(pk, r_1)|| \ldots ||\mathsf{F}(pk, r_n)$, $k = \mathcal{SS}.\mathsf{Rec}(r_i, r_i')$
   checks that $\Sigma.\mathsf{Verify}_{U_1}(\mathcal{U}||pk||r_2'|| \ldots ||r_n'||\mathsf{sid}_{U_i}, \sigma') = 1$
5. **Key Confirmation.** U_i, $i = 2, \ldots, n$:
   computes r_j s.t. $\mathcal{SS}.\mathsf{Share}(k, 2) = \{r_j, r_j'\}$, $j = 2, \ldots n$, $j \neq i$
   checks that $\mathsf{F}(pk, r_j)$ equals the one sent in Round 2

Olimid GKA

1. **Round 1.** U_1:
   runs $\mathcal{F}$.Gen to obtain (pk, sk), $r_1 \leftarrow^R \{0,1\}^{\mathcal{K}}$
   $U_1 \rightarrow^*$: $(\mathcal{U}, pk, \mathsf{F}(pk, r_1), \sigma = \Sigma.\mathsf{Sign}_{U_1}(\mathcal{U}||pk||\mathsf{F}(pk, r_1)))$
2. **Round 2.** U_i, $i = 2, \ldots, n$:
   checks that $\Sigma.\mathsf{Verify}_{U_1}(\mathcal{U}||pk||\mathsf{F}(pk, r_1), \sigma) = 1$, $r_i \leftarrow^R \{0,1\}^{\mathcal{K}}$
   $U_i \rightarrow^*$: $(\mathsf{F}(pk, r_i), \sigma_i = \Sigma.\mathsf{Sign}_{U_i}(\mathcal{U}||pk||\mathsf{F}(pk, r_i)))$
3. **Round 3.** U_1:
   checks that $\Sigma.\mathsf{Verify}_{U_i}(\mathcal{U}||pk||\mathsf{F}(pk, r_i), \sigma_i) = 1$, $i = 2, \ldots, n$
   computes $\mathsf{sid}_{U_1} = \mathsf{F}(pk, r_1)|| \ldots ||\mathsf{F}(pk, r_n)$, $r_i = \mathsf{F}^{-1}(sk, \mathsf{F}(pk, r_i))$
   $k = \mathsf{H}(\mathsf{sid}_{U_1}||r_1||r_2|| \ldots ||r_n)$, r_i' s.t. $\mathcal{SS}.\mathsf{Share}(k, 2) = \{r_i, r_i'\}$
   $U_1 \rightarrow^*$: $(r_2', \ldots r_n', \sigma' = \Sigma.\mathsf{Sign}_{U_1}(\mathcal{U}||pk||r_2'|| \ldots ||r_n'||\mathsf{sid}_{U_1}))$
4. **Key Computation.** U_i, $i = 2, \ldots, n$:
   checks that $\Sigma.\mathsf{Verify}_{U_j}(\mathcal{U}||pk||\mathsf{F}(pk, r_j), \sigma_j) = 1$, $j = 2, \ldots n$, $j \neq i$
   computes $\mathsf{sid}_{U_i} = \mathsf{F}(pk, r_1)|| \ldots ||\mathsf{F}(pk, r_n)$, $k = \mathcal{SS}.\mathsf{Rec}(r_i, r_i')$
   checks that $\Sigma.\mathsf{Verify}_{U_1}(\mathcal{U}||pk||r_2'|| \ldots ||r_n'||\mathsf{sid}_{U_i}, \sigma') = 1$
5. **Key Confirmation.** U_i, $i = 2, \ldots, n$:
   computes r_j s.t. $\mathcal{SS}.\mathsf{Share}(k, 2) = \{r_j, r_j'\}$, $j = 2, \ldots n$, $j \neq i$
   checks that $\mathsf{F}(pk, r_j)$ equals the one sent in Round 2

Olimid GKA

1. **Round 1.** U_1:
   runs $\mathcal{F}$.Gen to obtain (pk, sk), $r_1 \leftarrow^R \{0,1\}^{\mathcal{K}}$
   $U_1 \rightarrow^*$: $(\mathcal{U}, pk, \mathsf{F}(pk, r_1), \sigma = \Sigma.\mathsf{Sign}_{U_1}(\mathcal{U}||pk||\mathsf{F}(pk, r_1)))$
2. **Round 2.** U_i, $i = 2, \ldots, n$:
   checks that $\Sigma.\mathsf{Verify}_{U_1}(\mathcal{U}||pk||\mathsf{F}(pk, r_1), \sigma) = 1$, $r_i \leftarrow^R \{0,1\}^{\mathcal{K}}$
   $U_i \rightarrow^*$: $(\mathsf{F}(pk, r_i), \sigma_i = \Sigma.\mathsf{Sign}_{U_i}(\mathcal{U}||pk||\mathsf{F}(pk, r_i)))$
3. **Round 3.** U_1:
   checks that $\Sigma.\mathsf{Verify}_{U_i}(\mathcal{U}||pk||\mathsf{F}(pk, r_i), \sigma_i) = 1$, $i = 2, \ldots, n$
   computes $\mathsf{sid}_{U_1} = \mathsf{F}(pk, r_1)|| \ldots ||\mathsf{F}(pk, r_n)$, $r_i = \mathsf{F}^{-1}(sk, \mathsf{F}(pk, r_i))$
   $k = \mathsf{H}(\mathsf{sid}_{U_1}||r_1||r_2|| \ldots ||r_n)$, r_i' s.t. $\mathcal{SS}.\mathsf{Share}(k, 2) = \{r_i, r_i'\}$
   $U_1 \rightarrow^*$: $(r_2', \ldots r_n', \sigma' = \Sigma.\mathsf{Sign}_{U_1}(\mathcal{U}||pk||r_2'|| \ldots ||r_n'||\mathsf{sid}_{U_1}))$
4. **Key Computation.** U_i, $i = 2, \ldots, n$:
   checks that $\Sigma.\mathsf{Verify}_{U_j}(\mathcal{U}||pk||\mathsf{F}(pk, r_j), \sigma_j) = 1$, $j = 2, \ldots n$, $j \neq i$
   computes $\mathsf{sid}_{U_i} = \mathsf{F}(pk, r_1)|| \ldots ||\mathsf{F}(pk, r_n)$, $k = \mathcal{SS}.\mathsf{Rec}(r_i, r_i')$
   checks that $\Sigma.\mathsf{Verify}_{U_1}(\mathcal{U}||pk||r_2'|| \ldots ||r_n'||\mathsf{sid}_{U_i}, \sigma') = 1$
5. **Key Confirmation.** U_i, $i = 2, \ldots, n$:
   computes r_j s.t. $\mathcal{SS}.\mathsf{Share}(k, 2) = \{r_j, r_j'\}$, $j = 2, \ldots n$, $j \neq i$
   checks that $\mathsf{F}(pk, r_j)$ equals the one sent in Round 2

Olimid GKA

1. **Round 1.** U_1:
   runs $\mathcal{F}$.Gen to obtain (pk, sk), $r_1 \leftarrow^R \{0,1\}^{\mathcal{K}}$
   $U_1 \rightarrow^*$: $(\mathcal{U}, pk, \mathsf{F}(pk, r_1), \sigma = \Sigma.\mathsf{Sign}_{U_1}(\mathcal{U}||pk||\mathsf{F}(pk, r_1)))$
2. **Round 2.** U_i, $i = 2, \ldots, n$:
   checks that $\Sigma.\mathsf{Verify}_{U_1}(\mathcal{U}||pk||\mathsf{F}(pk, r_1), \sigma) = 1$, $r_i \leftarrow^R \{0,1\}^{\mathcal{K}}$
   $U_i \rightarrow^*$: $(\mathsf{F}(pk, r_i), \sigma_i = \Sigma.\mathsf{Sign}_{U_i}(\mathcal{U}||pk||\mathsf{F}(pk, r_i)))$
3. **Round 3.** U_1:
   checks that $\Sigma.\mathsf{Verify}_{U_i}(\mathcal{U}||pk||\mathsf{F}(pk, r_i), \sigma_i) = 1$, $i = 2, \ldots, n$
   computes $\mathsf{sid}_{U_1} = \mathsf{F}(pk, r_1)|| \ldots ||\mathsf{F}(pk, r_n)$, $r_i = \mathsf{F}^{-1}(sk, \mathsf{F}(pk, r_i))$
   $k = \mathsf{H}(\mathsf{sid}_{U_1}||r_1||r_2|| \ldots ||r_n)$, r_i' s.t. $\mathcal{SS}.\mathsf{Share}(k, 2) = \{r_i, r_i'\}$
   $U_1 \rightarrow^*$: $(r_2', \ldots r_n', \sigma' = \Sigma.\mathsf{Sign}_{U_1}(\mathcal{U}||pk||r_2'|| \ldots ||r_n'||\mathsf{sid}_{U_1}))$
4. **Key Computation.** U_i, $i = 2, \ldots, n$:
   checks that $\Sigma.\mathsf{Verify}_{U_j}(\mathcal{U}||pk||\mathsf{F}(pk, r_j), \sigma_j) = 1$, $j = 2, \ldots n$, $j \neq i$
   computes $\mathsf{sid}_{U_i} = \mathsf{F}(pk, r_1)|| \ldots ||\mathsf{F}(pk, r_n)$, $k = \mathcal{SS}.\mathsf{Rec}(r_i, r_i')$
   checks that $\Sigma.\mathsf{Verify}_{U_1}(\mathcal{U}||pk||r_2'|| \ldots ||r_n'||\mathsf{sid}_{U_i}, \sigma') = 1$
5. **Key Confirmation.** U_i, $i = 2, \ldots, n$:
   computes r_j s.t. $\mathcal{SS}.\mathsf{Share}(k, 2) = \{r_j, r_j'\}$, $j = 2, \ldots n$, $j \neq i$
   checks that $\mathsf{F}(pk, r_j)$ equals the one sent in Round 2

Olimid GKA

1. **Round 1.** U_1:
   runs $\mathcal{F}$.Gen to obtain (pk, sk), $r_1 \leftarrow^R \{0,1\}^{\mathcal{K}}$
   $U_1 \rightarrow^*$: $(\mathcal{U}, pk, \mathsf{F}(pk, r_1), \sigma = \Sigma.\mathsf{Sign}_{U_1}(\mathcal{U}||pk||\mathsf{F}(pk, r_1)))$
2. **Round 2.** U_i, $i = 2, \ldots, n$:
   checks that $\Sigma.\mathsf{Verify}_{U_1}(\mathcal{U}||pk||\mathsf{F}(pk, r_1), \sigma) = 1$, $r_i \leftarrow^R \{0,1\}^{\mathcal{K}}$
   $U_i \rightarrow^*$: $(\mathsf{F}(pk, r_i), \sigma_i = \Sigma.\mathsf{Sign}_{U_i}(\mathcal{U}||pk||\mathsf{F}(pk, r_i)))$
3. **Round 3.** U_1:
   checks that $\Sigma.\mathsf{Verify}_{U_i}(\mathcal{U}||pk||\mathsf{F}(pk, r_i), \sigma_i) = 1$, $i = 2, \ldots, n$
   computes $\mathsf{sid}_{U_1} = \mathsf{F}(pk, r_1)||\ldots||\mathsf{F}(pk, r_n)$, $r_i = \mathsf{F}^{-1}(sk, \mathsf{F}(pk, r_i))$
   $k = \mathsf{H}(\mathsf{sid}_{U_1}||r_1||r_2||\ldots||r_n)$, r_i' s.t. $\mathcal{SS}.\mathsf{Share}(k, 2) = \{r_i, r_i'\}$
   $U_1 \rightarrow^*$: $(r_2', \ldots r_n', \sigma' = \Sigma.\mathsf{Sign}_{U_1}(\mathcal{U}||pk||r_2'||\ldots||r_n'||\mathsf{sid}_{U_1}))$
4. **Key Computation.** U_i, $i = 2, \ldots, n$:
   checks that $\Sigma.\mathsf{Verify}_{U_j}(\mathcal{U}||pk||\mathsf{F}(pk, r_j), \sigma_j) = 1$, $j = 2, \ldots n$, $j \neq i$
   computes $\mathsf{sid}_{U_i} = \mathsf{F}(pk, r_1)||\ldots||\mathsf{F}(pk, r_n)$, $k = \mathcal{SS}.\mathsf{Rec}(r_i, r_i')$
   checks that $\Sigma.\mathsf{Verify}_{U_1}(\mathcal{U}||pk||r_2'||\ldots||r_n'||\mathsf{sid}_{U_i}, \sigma') = 1$
5. **Key Confirmation.** U_i, $i = 2, \ldots, n$:
   computes r_j s.t. $\mathcal{SS}.\mathsf{Share}(k, 2) = \{r_j, r_j'\}$, $j = 2, \ldots n$, $j \neq i$
   checks that $\mathsf{F}(pk, r_j)$ equals the one sent in Round 2

Olimid GKA

1. **Round 1.** U_1:
   runs $\mathcal{F}$.Gen to obtain (pk, sk), $r_1 \leftarrow^R \{0,1\}^{\mathcal{K}}$
   $U_1 \rightarrow^*$: $(\mathcal{U}, pk, \mathsf{F}(pk, r_1), \sigma = \Sigma.\mathsf{Sign}_{U_1}(\mathcal{U}||pk||\mathsf{F}(pk, r_1)))$
2. **Round 2.** U_i, $i = 2, \ldots, n$:
   checks that $\Sigma.\mathsf{Verify}_{U_1}(\mathcal{U}||pk||\mathsf{F}(pk, r_1), \sigma) = 1$, $r_i \leftarrow^R \{0,1\}^{\mathcal{K}}$
   $U_i \rightarrow^*$: $(\mathsf{F}(pk, r_i), \sigma_i = \Sigma.\mathsf{Sign}_{U_i}(\mathcal{U}||pk||\mathsf{F}(pk, r_i)))$
3. **Round 3.** U_1:
   checks that $\Sigma.\mathsf{Verify}_{U_i}(\mathcal{U}||pk||\mathsf{F}(pk, r_i), \sigma_i) = 1$, $i = 2, \ldots, n$
   computes $\mathsf{sid}_{U_1} = \mathsf{F}(pk, r_1)|| \ldots ||\mathsf{F}(pk, r_n)$, $r_i = \mathsf{F}^{-1}(sk, \mathsf{F}(pk, r_i))$
   $k = \mathsf{H}(\mathsf{sid}_{U_1}||r_1||r_2|| \ldots ||r_n)$, r_i' s.t. $\mathcal{SS}.\mathsf{Share}(k, 2) = \{r_i, r_i'\}$
   $U_1 \rightarrow^*$: $(r_2', \ldots r_n', \sigma' = \Sigma.\mathsf{Sign}_{U_1}(\mathcal{U}||pk||r_2'|| \ldots ||r_n'||\mathsf{sid}_{U_1}))$
4. **Key Computation.** U_i, $i = 2, \ldots, n$:
   checks that $\Sigma.\mathsf{Verify}_{U_j}(\mathcal{U}||pk||\mathsf{F}(pk, r_j), \sigma_j) = 1$, $j = 2, \ldots n$, $j \neq i$
   computes $\mathsf{sid}_{U_i} = \mathsf{F}(pk, r_1)|| \ldots ||\mathsf{F}(pk, r_n)$, $k = \mathcal{SS}.\mathsf{Rec}(r_i, r_i')$
   checks that $\Sigma.\mathsf{Verify}_{U_1}(\mathcal{U}||pk||r_2'|| \ldots ||r_n'||\mathsf{sid}_{U_i}, \sigma') = 1$
5. **Key Confirmation.** U_i, $i = 2, \ldots, n$:
   computes r_j s.t. $\mathcal{SS}.\mathsf{Share}(k, 2) = \{r_j, r_j'\}$, $j = 2, \ldots n$, $j \neq i$
   checks that $\mathsf{F}(pk, r_j)$ equals the one sent in Round 2

Olimid GKA

1. **Round 1.** U_1:
   runs $\mathcal{F}$.Gen to obtain (pk, sk), $r_1 \leftarrow^R \{0,1\}^{\mathcal{K}}$
   $U_1 \rightarrow^*$: $(\mathcal{U}, pk, \mathsf{F}(pk, r_1), \sigma = \Sigma.\mathsf{Sign}_{U_1}(\mathcal{U}||pk||\mathsf{F}(pk, r_1)))$
2. **Round 2.** U_i, $i = 2, \ldots, n$:
   checks that $\Sigma.\mathsf{Verify}_{U_1}(\mathcal{U}||pk||\mathsf{F}(pk, r_1), \sigma) = 1$, $r_i \leftarrow^R \{0,1\}^{\mathcal{K}}$
   $U_i \rightarrow^*$: $(\mathsf{F}(pk, r_i), \sigma_i = \Sigma.\mathsf{Sign}_{U_i}(\mathcal{U}||pk||\mathsf{F}(pk, r_i)))$
3. **Round 3.** U_1:
   checks that $\Sigma.\mathsf{Verify}_{U_i}(\mathcal{U}||pk||\mathsf{F}(pk, r_i), \sigma_i) = 1$, $i = 2, \ldots, n$
   computes $\mathsf{sid}_{U_1} = \mathsf{F}(pk, r_1)|| \ldots ||\mathsf{F}(pk, r_n)$, $r_i = \mathsf{F}^{-1}(sk, \mathsf{F}(pk, r_i))$
   $k = \mathsf{H}(\mathsf{sid}_{U_1}||r_1||r_2|| \ldots ||r_n)$, r_i' s.t. $\mathcal{SS}.\mathsf{Share}(k, 2) = \{r_i, r_i'\}$
   $U_1 \rightarrow^*$: $(r_2', \ldots r_n', \sigma' = \Sigma.\mathsf{Sign}_{U_1}(\mathcal{U}||pk||r_2'|| \ldots ||r_n'||\mathsf{sid}_{U_1}))$
4. **Key Computation.** U_i, $i = 2, \ldots, n$:
   checks that $\Sigma.\mathsf{Verify}_{U_j}(\mathcal{U}||pk||\mathsf{F}(pk, r_j), \sigma_j) = 1$, $j = 2, \ldots n$, $j \neq i$
   computes $\mathsf{sid}_{U_i} = \mathsf{F}(pk, r_1)|| \ldots ||\mathsf{F}(pk, r_n)$, $k = \mathcal{SS}.\mathsf{Rec}(r_i, r_i')$
   checks that $\Sigma.\mathsf{Verify}_{U_1}(\mathcal{U}||pk||r_2'|| \ldots ||r_n'||\mathsf{sid}_{U_i}, \sigma') = 1$
5. **Key Confirmation.** U_i, $i = 2, \ldots, n$:
   computes r_j s.t. $\mathcal{SS}.\mathsf{Share}(k, 2) = \{r_j, r_j'\}$, $j = 2, \ldots n$, $j \neq i$
   checks that $\mathsf{F}(pk, r_j)$ equals the one sent in Round 2

Olimid GKA

1. **Round 1.** U_1:
   runs $\mathcal{F}$.Gen to obtain (pk, sk), $r_1 \leftarrow^R \{0,1\}^{\mathcal{K}}$
   $U_1 \rightarrow^*$: $(\mathcal{U}, pk, \mathsf{F}(pk, r_1), \sigma = \Sigma.\mathsf{Sign}_{U_1}(\mathcal{U}||pk||\mathsf{F}(pk, r_1)))$
2. **Round 2.** U_i, $i = 2, \ldots, n$:
   checks that $\Sigma.\mathsf{Verify}_{U_1}(\mathcal{U}||pk||\mathsf{F}(pk, r_1), \sigma) = 1$, $r_i \leftarrow^R \{0,1\}^{\mathcal{K}}$
   $U_i \rightarrow^*$: $(\mathsf{F}(pk, r_i), \sigma_i = \Sigma.\mathsf{Sign}_{U_i}(\mathcal{U}||pk||\mathsf{F}(pk, r_i)))$
3. **Round 3.** U_1:
   checks that $\Sigma.\mathsf{Verify}_{U_i}(\mathcal{U}||pk||\mathsf{F}(pk, r_i), \sigma_i) = 1$, $i = 2, \ldots, n$
   computes $\mathsf{sid}_{U_1} = \mathsf{F}(pk, r_1)|| \ldots ||\mathsf{F}(pk, r_n)$, $r_i = \mathsf{F}^{-1}(sk, \mathsf{F}(pk, r_i))$
   $k = \mathsf{H}(\mathsf{sid}_{U_1}||r_1||r_2|| \ldots ||r_n)$, r_i' s.t. $\mathcal{SS}.\mathsf{Share}(k, 2) = \{r_i, r_i'\}$
   $U_1 \rightarrow^*$: $(r_2', \ldots r_n', \sigma' = \Sigma.\mathsf{Sign}_{U_1}(\mathcal{U}||pk||r_2'|| \ldots ||r_n'||\mathsf{sid}_{U_1}))$
4. **Key Computation.** U_i, $i = 2, \ldots, n$:
   checks that $\Sigma.\mathsf{Verify}_{U_j}(\mathcal{U}||pk||\mathsf{F}(pk, r_j), \sigma_j) = 1$, $j = 2, \ldots n$, $j \neq i$
   computes $\mathsf{sid}_{U_i} = \mathsf{F}(pk, r_1)|| \ldots ||\mathsf{F}(pk, r_n)$, $k = \mathcal{SS}.\mathsf{Rec}(r_i, r_i')$
   checks that $\Sigma.\mathsf{Verify}_{U_1}(\mathcal{U}||pk||r_2'|| \ldots ||r_n'||\mathsf{sid}_{U_i}, \sigma') = 1$
5. **Key Confirmation.** U_i, $i = 2, \ldots, n$:
   computes r_j s.t. $\mathcal{SS}.\mathsf{Share}(k, 2) = \{r_j, r_j'\}$, $j = 2, \ldots n$, $j \neq i$
   checks that $\mathsf{F}(pk, r_j)$ equals the one sent in Round 2

Olimid GKA

1. **Round 1.** U_1:
   runs $\mathcal{F}$.Gen to obtain (pk, sk), $r_1 \leftarrow^R \{0,1\}^{\mathcal{K}}$
   $U_1 \rightarrow^*$: $(\mathcal{U}, pk, \mathsf{F}(pk, r_1), \sigma = \Sigma.\mathsf{Sign}_{U_1}(\mathcal{U}||pk||\mathsf{F}(pk, r_1)))$
2. **Round 2.** U_i, $i = 2, \ldots, n$:
   checks that $\Sigma.\mathsf{Verify}_{U_1}(\mathcal{U}||pk||\mathsf{F}(pk, r_1), \sigma) = 1$, $r_i \leftarrow^R \{0,1\}^{\mathcal{K}}$
   $U_i \rightarrow^*$: $(\mathsf{F}(pk, r_i), \sigma_i = \Sigma.\mathsf{Sign}_{U_i}(\mathcal{U}||pk||\mathsf{F}(pk, r_i)))$
3. **Round 3.** U_1:
   checks that $\Sigma.\mathsf{Verify}_{U_i}(\mathcal{U}||pk||\mathsf{F}(pk, r_i), \sigma_i) = 1$, $i = 2, \ldots, n$
   computes $\mathsf{sid}_{U_1} = \mathsf{F}(pk, r_1)|| \ldots ||\mathsf{F}(pk, r_n)$, $r_i = \mathsf{F}^{-1}(sk, \mathsf{F}(pk, r_i))$
   $k = \mathsf{H}(\mathsf{sid}_{U_1}||r_1||r_2|| \ldots ||r_n)$, r_i' s.t. $\mathcal{SS}.\mathsf{Share}(k, 2) = \{r_i, r_i'\}$
   $U_1 \rightarrow^*$: $(r_2', \ldots r_n', \sigma' = \Sigma.\mathsf{Sign}_{U_1}(\mathcal{U}||pk||r_2'|| \ldots ||r_n'||\mathsf{sid}_{U_1}))$
4. **Key Computation.** U_i, $i = 2, \ldots, n$:
   checks that $\Sigma.\mathsf{Verify}_{U_j}(\mathcal{U}||pk||\mathsf{F}(pk, r_j), \sigma_j) = 1$, $j = 2, \ldots n$, $j \neq i$
   computes $\mathsf{sid}_{U_i} = \mathsf{F}(pk, r_1)|| \ldots ||\mathsf{F}(pk, r_n)$, $k = \mathcal{SS}.\mathsf{Rec}(r_i, r_i')$
   checks that $\Sigma.\mathsf{Verify}_{U_1}(\mathcal{U}||pk||r_2'|| \ldots ||r_n'||\mathsf{sid}_{U_i}, \sigma') = 1$
5. **Key Confirmation.** U_i, $i = 2, \ldots, n$:
   computes r_j s.t. $\mathcal{SS}.\mathsf{Share}(k, 2) = \{r_j, r_j'\}$, $j = 2, \ldots n$, $j \neq i$
   checks that $\mathsf{F}(pk, r_j)$ equals the one sent in Round 2

Olimid GKA

1. **Round 1.** U_1:
   runs $\mathcal{F}$.Gen to obtain (pk, sk), $r_1 \leftarrow^R \{0,1\}^{\mathcal{K}}$
   $U_1 \rightarrow^*$: $(\mathcal{U}, pk, \mathsf{F}(pk, r_1), \sigma = \Sigma.\mathsf{Sign}_{U_1}(\mathcal{U}||pk||\mathsf{F}(pk, r_1)))$
2. **Round 2.** U_i, $i = 2, \ldots, n$:
   checks that $\Sigma.\mathsf{Verify}_{U_1}(\mathcal{U}||pk||\mathsf{F}(pk, r_1), \sigma) = 1$, $r_i \leftarrow^R \{0,1\}^{\mathcal{K}}$
   $U_i \rightarrow^*$: $(\mathsf{F}(pk, r_i), \sigma_i = \Sigma.\mathsf{Sign}_{U_i}(\mathcal{U}||pk||\mathsf{F}(pk, r_i)))$
3. **Round 3.** U_1:
   checks that $\Sigma.\mathsf{Verify}_{U_i}(\mathcal{U}||pk||\mathsf{F}(pk, r_i), \sigma_i) = 1$, $i = 2, \ldots, n$
   computes $\mathsf{sid}_{U_1} = \mathsf{F}(pk, r_1)|| \ldots ||\mathsf{F}(pk, r_n)$, $r_i = \mathsf{F}^{-1}(sk, \mathsf{F}(pk, r_i))$
   $k = \mathsf{H}(\mathsf{sid}_{U_1}||r_1||r_2|| \ldots ||r_n)$, r_i' s.t. $\mathcal{SS}.\mathsf{Share}(k, 2) = \{r_i, r_i'\}$
   $U_1 \rightarrow^*$: $(r_2', \ldots r_n', \sigma' = \Sigma.\mathsf{Sign}_{U_1}(\mathcal{U}||pk||r_2'|| \ldots ||r_n'||\mathsf{sid}_{U_1}))$
4. **Key Computation.** U_i, $i = 2, \ldots, n$:
   checks that $\Sigma.\mathsf{Verify}_{U_j}(\mathcal{U}||pk||\mathsf{F}(pk, r_j), \sigma_j) = 1$, $j = 2, \ldots n$, $j \neq i$
   computes $\mathsf{sid}_{U_i} = \mathsf{F}(pk, r_1)|| \ldots ||\mathsf{F}(pk, r_n)$, $k = \mathcal{SS}.\mathsf{Rec}(r_i, r_i')$
   checks that $\Sigma.\mathsf{Verify}_{U_1}(\mathcal{U}||pk||r_2'|| \ldots ||r_n'||\mathsf{sid}_{U_i}, \sigma') = 1$
5. **Key Confirmation.** U_i, $i = 2, \ldots, n$:
   computes r_j s.t. $\mathcal{SS}.\mathsf{Share}(k, 2) = \{r_j, r_j'\}$, $j = 2, \ldots n$, $j \neq i$
   checks that $\mathsf{F}(pk, r_j)$ equals the one sent in Round 2

Olimid GKA

1. **Round 1.** U_1:
   runs $\mathcal{F}$.Gen to obtain (pk, sk), $r_1 \leftarrow^R \{0,1\}^{\mathcal{K}}$
   $U_1 \rightarrow^*$: $(\mathcal{U}, pk, \mathsf{F}(pk, r_1), \sigma = \Sigma.\mathsf{Sign}_{U_1}(\mathcal{U}||pk||\mathsf{F}(pk, r_1)))$
2. **Round 2.** U_i, $i = 2, \ldots, n$:
   checks that $\Sigma.\mathsf{Verify}_{U_1}(\mathcal{U}||pk||\mathsf{F}(pk, r_1), \sigma) = 1$, $r_i \leftarrow^R \{0,1\}^{\mathcal{K}}$
   $U_i \rightarrow^*$: $(\mathsf{F}(pk, r_i), \sigma_i = \Sigma.\mathsf{Sign}_{U_i}(\mathcal{U}||pk||\mathsf{F}(pk, r_i)))$
3. **Round 3.** U_1:
   checks that $\Sigma.\mathsf{Verify}_{U_i}(\mathcal{U}||pk||\mathsf{F}(pk, r_i), \sigma_i) = 1$, $i = 2, \ldots, n$
   computes $\mathsf{sid}_{U_1} = \mathsf{F}(pk, r_1)|| \ldots ||\mathsf{F}(pk, r_n)$, $r_i = \mathsf{F}^{-1}(sk, \mathsf{F}(pk, r_i))$
   $k = \mathsf{H}(\mathsf{sid}_{U_1}||r_1||r_2|| \ldots ||r_n)$, r_i' s.t. $\mathcal{SS}.\mathsf{Share}(k, 2) = \{r_i, r_i'\}$
   $U_1 \rightarrow^*$: $(r_2', \ldots r_n', \sigma' = \Sigma.\mathsf{Sign}_{U_1}(\mathcal{U}||pk||r_2'|| \ldots ||r_n'||\mathsf{sid}_{U_1}))$
4. **Key Computation.** U_i, $i = 2, \ldots, n$:
   checks that $\Sigma.\mathsf{Verify}_{U_j}(\mathcal{U}||pk||\mathsf{F}(pk, r_j), \sigma_j) = 1$, $j = 2, \ldots n$, $j \neq i$
   computes $\mathsf{sid}_{U_i} = \mathsf{F}(pk, r_1)|| \ldots ||\mathsf{F}(pk, r_n)$, $k = \mathcal{SS}.\mathsf{Rec}(r_i, r_i')$
   checks that $\Sigma.\mathsf{Verify}_{U_1}(\mathcal{U}||pk||r_2'|| \ldots ||r_n'||\mathsf{sid}_{U_i}, \sigma') = 1$
5. **Key Confirmation.** U_i, $i = 2, \ldots, n$:
   computes r_j s.t. $\mathcal{SS}.\mathsf{Share}(k, 2) = \{r_j, r_j'\}$, $j = 2, \ldots n$, $j \neq i$
   checks that $\mathsf{F}(pk, r_j)$ equals the one sent in Round 2

Olimid GKA

1. **Round 1.** U_1:
   runs $\mathcal{F}$.Gen to obtain (pk, sk), $r_1 \leftarrow^R \{0,1\}^{\mathcal{K}}$
   $U_1 \rightarrow^*$: $(\mathcal{U}, pk, \mathsf{F}(pk, r_1), \sigma = \Sigma.\mathsf{Sign}_{U_1}(\mathcal{U}||pk||\mathsf{F}(pk, r_1)))$
2. **Round 2.** U_i, $i = 2, \ldots, n$:
   checks that $\Sigma.\mathsf{Verify}_{U_1}(\mathcal{U}||pk||\mathsf{F}(pk, r_1), \sigma) = 1$, $r_i \leftarrow^R \{0,1\}^{\mathcal{K}}$
   $U_i \rightarrow^*$: $(\mathsf{F}(pk, r_i), \sigma_i = \Sigma.\mathsf{Sign}_{U_i}(\mathcal{U}||pk||\mathsf{F}(pk, r_i)))$
3. **Round 3.** U_1:
   checks that $\Sigma.\mathsf{Verify}_{U_i}(\mathcal{U}||pk||\mathsf{F}(pk, r_i), \sigma_i) = 1$, $i = 2, \ldots, n$
   computes $\mathsf{sid}_{U_1} = \mathsf{F}(pk, r_1)|| \ldots ||\mathsf{F}(pk, r_n)$, $r_i = \mathsf{F}^{-1}(sk, \mathsf{F}(pk, r_i))$
   $k = \mathsf{H}(\mathsf{sid}_{U_1}||r_1||r_2|| \ldots ||r_n)$, r_i' s.t. $\mathcal{SS}.\mathsf{Share}(k, 2) = \{r_i, r_i'\}$
   $U_1 \rightarrow^*$: $(r_2', \ldots r_n', \sigma' = \Sigma.\mathsf{Sign}_{U_1}(\mathcal{U}||pk||r_2'|| \ldots ||r_n'||\mathsf{sid}_{U_1}))$
4. **Key Computation.** U_i, $i = 2, \ldots, n$:
   checks that $\Sigma.\mathsf{Verify}_{U_j}(\mathcal{U}||pk||\mathsf{F}(pk, r_j), \sigma_j) = 1$, $j = 2, \ldots n$, $j \neq i$
   computes $\mathsf{sid}_{U_i} = \mathsf{F}(pk, r_1)|| \ldots ||\mathsf{F}(pk, r_n)$, $k = \mathcal{SS}.\mathsf{Rec}(r_i, r_i')$
   checks that $\Sigma.\mathsf{Verify}_{U_1}(\mathcal{U}||pk||r_2'|| \ldots ||r_n'||\mathsf{sid}_{U_i}, \sigma') = 1$
5. **Key Confirmation.** U_i, $i = 2, \ldots, n$:
   computes r_j s.t. $\mathcal{SS}.\mathsf{Share}(k, 2) = \{r_j, r_j'\}$, $j = 2, \ldots n$, $j \neq i$
   checks that $\mathsf{F}(pk, r_j)$ equals the one sent in Round 2

GBG Model

- **G**orantla, **B**oyd, **G**onzalez Nieto, 2009

GBG Model

- **G**orantla, **B**oyd, **G**onzalez Nieto, 2009
- Models:
  - sessions: sid_U^s, upper bounded by q_s
  - users: Π_U^s, may accept K_U^s, own a public-private *long-term key* pair (pk_i, sk_i), maintain internal states
  - attacker: PPT, active, asks queries (Execute, Send, RevealKey, RevealState, Corrupt, Test)

GBG Model

- **G**orantla, **B**oyd, **G**onzalez Nieto, 2009
- Models:
  - sessions: sid_U^s, upper bounded by q_s
  - users: Π_U^s, may accept K_U^s, own a public-private *long-term key* pair (pk_i, sk_i), maintain internal states
  - attacker: PPT, active, asks queries (Execute, Send, RevealKey, RevealState, Corrupt, Test)
- Security Notions:
  - AKE: *key confidentiality, forward secrecy, known key security, ...*
  - MA: *mutual authentication, key confirmation, key compromise impersonation resilience, ...*
  - *Contributiveness: equal contribution of the parties to the key establishment, key freshness, key unpredictability, key randomness, ...*

Security Proofs I

Theorem 1 (AKE Security)

If the signature scheme Σ is UF-CMA, the trapdoor function $\mathcal{F}$ is secure, the hash function H is a random oracle and the secret n-sharing scheme $\mathcal{SS}$ is perfect then our protocol is AKE secure and

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{AKE}} \leq 2n^2\mathsf{Adv}_{\mathcal{A},\Sigma}^{\mathsf{UF-CMA}} + \frac{(q_s+q_r)^2}{2^{k-1}} + \frac{(n+1){q_s}^2}{2^{k-1}} + 2\mathsf{Adv}_{\mathcal{A},\mathcal{F}}.$$

Security Proofs II

Theorem 2 (MA Security)

If the signature scheme Σ is UF-CMA and the hash function H *is a random oracle then our protocol is MA secure and*

$$\mathrm{Adv}_{\mathcal{A}}^{\mathrm{MA}} \leq n^2 \mathrm{Adv}_{\mathcal{A},\Sigma}^{\mathrm{UF-CMA}} + \frac{(q_s + q_r)^2}{2^k} + \frac{(n+1){q_s}^2}{2^k}.$$

Theorem 3 (Contributiveness)

If the trapdoor function $\mathcal{F}$ is secure and the hash function H *is a random oracle then our protocol is contributive and*

$$\mathrm{Adv}_{\mathcal{A}}^{\mathrm{Con}} \leq \frac{(n+1){q_s}^2}{2^k} + \frac{q_r}{2^k}.$$

Complexity Analysis

	Storage	Computation	Transmission
U_1	$l_{sk} + l_{sk_1} + k$	$c_{\mathcal{F}.\mathsf{G}} + c_{\mathsf{F}} + (n-1)c_{\mathsf{F}^{-1}} + c_{\mathsf{H}}$ $2c_{\Sigma.\mathsf{Sign}} + (n-1)c_{\Sigma.\mathsf{Verify}}$ $+(n-1)c_{\mathcal{SS}.\mathsf{Share}}$	$l_{\mathcal{U}} + l_{pk} + nl_{\mathsf{F}} +$ $+(n+1)l_{\Sigma.\mathsf{Sign}} +$ $+(n-1)k$
U_i $(i \neq 1)$	$l_{sk_i} + k$	$c_{\mathsf{F}} + c_{\Sigma.\mathsf{Sign}} + nc_{\Sigma.\mathsf{Verify}} + c_{\mathcal{SS}.\mathsf{Rec}}$ $(+(n-2)c_{\mathcal{SS}.\mathsf{Share}})$	

Comparison to existing work

	Bresson-Catalano	Cao et al.	Our Protocol
No.of Rounds	3	3	3
Trans.Type	unicast, broadcast	broadcast	broadcast
No.of Messages	$n(n-1), 2n$	3n	n+1
Group Type	static	static	static
sid Generation	in advance	in advance	at runtime
Security Model	BCP	UC	GBG

Summary & Future work

We proposed a new GKA protocol that:

Summary & Future work

We proposed a new GKA protocol that:

- relies on *perfect secret n-sharing*

Summary & Future work

We proposed a new GKA protocol that:

- relies on *perfect secret n-sharing*
- satisfies GBG security

Summary & Future work

We proposed a new GKA protocol that:

- relies on *perfect secret n-sharing*
- satisfies GBG security
- is round-constant (3 rounds)

Summary & Future work

We proposed a new GKA protocol that:

- relies on *perfect secret n-sharing*
- satisfies GBG security
- is round-constant (3 rounds)
- uses broadcast only

Summary & Future work

We proposed a new GKA protocol that:

- relies on *perfect secret n-sharing*
- satisfies GBG security
- is round-constant (3 rounds)
- uses broadcast only
- is efficient in the no. of messages

Summary & Future work

We proposed a new GKA protocol that:

- relies on *perfect secret n-sharing*
- satisfies GBG security
- is round-constant (3 rounds)
- uses broadcast only
- is efficient in the no. of messages
- allows static groups

Summary & Future work

We proposed a new GKA protocol that:

- relies on *perfect secret n-sharing*
- satisfies GBG security $\Rightarrow$ achieve eGBG security
- is round-constant (3 rounds)
- uses broadcast only
- is efficient in the no. of messages
- allows static groups

Summary & Future work

We proposed a new GKA protocol that:

- relies on *perfect secret n-sharing*
- satisfies GBG security $\Rightarrow$ achieve eGBG security
- is round-constant (3 rounds)
- uses broadcast only
- is efficient in the no. of messages $\Rightarrow$ decrease message length
- allows static groups

Summary & Future work

We proposed a new GKA protocol that:

- relies on *perfect secret n-sharing*
- satisfies GBG security $\Rightarrow$ achieve eGBG security
- is round-constant (3 rounds)
- uses broadcast only
- is efficient in the no. of messages $\Rightarrow$ decrease message length
- allows static groups $\Rightarrow$ allow dynamic groups

Summary & Future work

We proposed a new GKA protocol that:

- relies on *perfect secret n-sharing*
- satisfies GBG security $\Rightarrow$ achieve eGBG security
- is round-constant (3 rounds)
- uses broadcast only
- is efficient in the no. of messages $\Rightarrow$ decrease message length
- allows static groups $\Rightarrow$ allow dynamic groups

Define new efficient and secure secret sharing-based GKA protocols.

Thank you!

Q & A