Dolev-Yao-type Abstraction of Modular Exponentiation - the Cliques Case Study

Olivier Pereira and Jean-Jacques Quisquater
UCL Crypto Group
Belgium
{pereira,quisquater}@dice.ucl.ac.be

June 2004

The SA-GDH.2 Protocol

Cliques SA-GDH.2 protocol with three participants
[AST at CCS'98 and IEEE J-SAC'00]

$$
\begin{tikzcd}
M_1 & M_2 & M_3 \\
\bullet \arrow[r, "\alpha, \alpha^{r_1 K_{12}}, \alpha^{r_1 K_{13}}"] \arrow[dd, Rightarrow] & \bullet \arrow[d, Rightarrow] & \\
& \bullet \arrow[r, "\alpha^{r_2 K_{21}}, \alpha^{r_1 K_{12}}, \alpha^{r_1 r_2 K_{13} K_{23}}"] \arrow[d, Rightarrow] & \bullet \arrow[d, Rightarrow] \\
\bullet & \bullet \arrow[l, "\alpha^{r_2 r_3 K_{21} K_{31}}, \alpha^{r_1 r_3 K_{12} K_{32}}"'] & \bullet \arrow[l, "\alpha^{r_2 r_3 K_{21} K_{31}}, \alpha^{r_1 r_3 K_{12} K_{32}}"']
\end{tikzcd}
$$

- α is a public generator of a group $\mathcal{G}$ where the DDH problem is believed to be hard
- M_i generates a random key contribution r_i
- M_i and M_j share long-term key K_{ij} ($Pub = \alpha^{x_i}$, $Priv = x_i$)
- All participants can compute $\alpha^{r_1 r_2 r_3}$

Security Goals

SA-GDH.2 protocol with group $\mathrm{M} = \{M_1, M_2, M_3\}$

$$
\begin{tikzcd}
M_1 & M_2 & M_3 \\
\bullet \arrow[r, "{\alpha, \alpha^{r_1 K_{12}}, \alpha^{r_1 K_{13}}}"] \arrow[dd, Rightarrow] & \bullet \arrow[d, Rightarrow] & \\
 & \bullet \arrow[r, "{\alpha^{r_2 K_{21}}, \alpha^{r_1 K_{12}}, \alpha^{r_1 r_2 K_{13} K_{23}}}"] \arrow[d, Rightarrow] & \bullet \arrow[d, Rightarrow] \\
\bullet & \bullet \arrow[l, "{\alpha^{r_2 r_3 K_{21} K_{31}}, \alpha^{r_1 r_3 K_{12} K_{32}}}"'] & \bullet \arrow[l, "{\alpha^{r_2 r_3 K_{21} K_{31}}, \alpha^{r_1 r_3 K_{12} K_{32}}}"']
\end{tikzcd}
$$

Main security goal:

- *Implicit Key Authentication*: no party $M_I \notin \mathrm{M}$ should be able to obtain any participant's view of the group key

Security Analysis

What can we say about the security of these protocols?

- [Steiner & al. 96] provide security proof of the generic Cliques GDH protocols in the case of a passive adversary
- [AST] only provide "sketch proofs"
- These protocols do not appear to fit into any classical security framework

Adversary Model

Dolev-Yao-type Adversary

- controls the network
- can take part to some sessions (has long-term K_{Ij})
- can build messages in accordance with certain “symbolic” rules
- rules are defined in order to make the attacker able to perform the same operations as any honest user

Message Algebra

Our message algebra is defined as follows

- R: set of random private values generated during protocol execution
- K: set of long-term secrets shared between pairs of users
- P: abelian group freely generated from $\mathsf{R} \cup \mathsf{K}$
- G: isomorphic to P through **alphaexp** : $\mathsf{P} \rightarrow \mathsf{G}$

Remarks:

- **alphaexp**(p) usually denoted α^{p}
- $\mathcal{G}$ was cyclic and is represented by G which is infinite
- freeness implies that $\alpha^{r_1 r_2} \neq \alpha^{r_3}$, $\alpha^{r_1 K_{12}} \neq \alpha^{K_{23}}$, ...

Adversary Capabilities

Adversary message generation capabilities

- Adversary knows:
  - all elements of G he intercepted
  - all elements of R he generated
  - all elements of K he shares with other users
- He knows the subgroup of P freely generated from the elements of R and K he knows
- If he knows $p \in \mathsf{P}$ and $g \in \mathsf{G}$, he can generate g^p $(= \mathbf{alphaexp}(\mathbf{alphaexp}^{-1}(g) \cdot p))$

Adversary Goal

The SA-GDH.2 Protocol

$$\begin{tikzcd}
M_1 & M_2 & M_3 \\
\bullet \arrow[r, "{\alpha, \alpha^{r_1 K_{12}}, \alpha^{r_1 K_{13}}}"] \arrow[dd, Rightarrow] & \bullet \arrow[d, Rightarrow] & \\
& \bullet \arrow[r, "{\alpha^{r_2 K_{21}}, \alpha^{r_1 K_{12}}, \alpha^{r_1 r_2 K_{13} K_{23}}}"] \arrow[d, Rightarrow] & \bullet \arrow[d, Rightarrow] \\
\bullet & \bullet \arrow[l, "{\alpha^{r_2 r_3 K_{21} K_{31}}, \alpha^{r_1 r_3 K_{12} K_{32}}}"'] & \bullet \arrow[l, "{\alpha^{r_2 r_3 K_{21} K_{31}}, \alpha^{r_1 r_3 K_{12} K_{32}}}"']
\end{tikzcd}$$

Consider M_2 for instance.
Adversary goal is:

- to obtain a pair $(\alpha^x, \alpha^{x r_2 K_{12}^{-1} K_{23}^{-1}})$ (for any x)
- to replace $\alpha^{r_1 r_3 K_{12} K_{32}}$ with α^x

Adversary Attack Strategy

How can he do this?

- Use his (Dolev-Yao) arithmetic capabilities
- Use the *services* offered by honest users

Services:

- M_2 says: "Send me 3 elements of G, I will exponentiate the first of them with r_2K_{21} and the third of them with r_2K_{23}"

We say that M_2 provides the r_2K_{21}- and r_2K_{23}-services

- M_3 provides the r_3K_{31}- and r_3K_{32}-services
- M_1 says: "I will exponentiate α with r_1K_{12} and r_1K_{13}"
  This can be seen as a services with fixed input...

Attack against the SA-GDH.2 Protocol

First session: $\{M_1, M_2, M_I\}$

$$M_1 \xrightarrow{\alpha,\, \alpha^{r_1 K_{12}},\, \alpha^{r_1 K_{1I}}}$$

Second session: $\{M_I, M_2, M_3\}$

$$\xrightarrow{\alpha^{r_1},\, \alpha^{x},\, \alpha^{r_1 K_{12}}} M_2$$

$$\Downarrow$$

$$\bullet \xrightarrow{\alpha^{r_1 r_2' K_{2I}},\, \alpha^{x},\, \alpha^{r_1 r_2' K_{12} K_{23}}}$$

$$\xrightarrow{\alpha^{r_1 r_2'},\, \alpha^{r_1 r_2' K_{12} K_{23}},\, \alpha^{x}} M_3$$

$$\Downarrow$$

$$\xleftarrow{\alpha^{r_1 r_2' r_3' K_{3I}},\, \alpha^{r_1 r_2' r_3' K_{12} K_{23} K_{32}}} \bullet$$

Attack against the SA-GDH.2 Protocol

Third session: $\{M_1, M_2, M_3\}$

$$\xrightarrow{\alpha, \alpha^x, \alpha^{r_1 r_2' r_3'}} M_2$$

$$\Downarrow$$

$$\bullet \xrightarrow{\alpha^{r_2'' K_{21}}, \alpha^x, \alpha^{r_1 r_2' r_3' r_2'' K_{23}}}$$

$$\Downarrow$$

$$\bullet \xleftarrow{\alpha^x, \alpha^{r_1 r_2' r_3' K_{12} K_{23} K_{32}}}$$

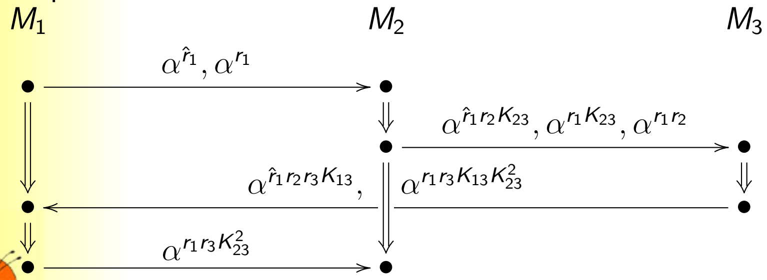
M_2 computes $\alpha^{r_1 r_2' r_3' r_2'' K_{23}}$ as group key even though the three group members simply followed the protocol definition!

How to fix this protocol?

We consider as a fix a protocol

- providing implicit key authentication (at least)
- allowing a group of n members to compute $\alpha^{r_1 \cdots r_n}$
- using the same "building blocks", i.e. exponentiation with random values and long-term two-party secrets

Example:



How to fix this protocol?

We consider as a fix a protocol

- providing implicit key authentication (at least)
- allowing a group of n members to compute $\alpha^{r_1 \cdots r_n}$
- using the same "building blocks", i.e. exponentiation with random values and long-term two-party secrets

Theorem:
This is impossible for protocols with at least 4 participants

Attack Process

First step:

- Find which services are to be used
- When trying to obtain $(\alpha^x, \alpha^{x r_2'' K_{12}^{-1} K_{32}^{-1}})$, look for a set of services and values the adversary knows whose product is $r_2'' K_{12}^{-1} K_{32}^{-1}$

Example:

$$\begin{aligned} r_2'' K_{12}^{-1} K_{32}^{-1} &= (r_1 K_{12})^{-1} \cdot r_1 K_{1I} \cdot K_{1I}^{-1} \cdot \\ &\quad (r_2' K_{23})^{-1} \cdot r_2' K_{2I} \cdot K_{2I}^{-1} \cdot \\ &\quad (r_3' K_{32})^{-1} \cdot r_3' K_{3I} \cdot K_{3I}^{-1} \cdot \\ &\quad r_2'' K_{23} \end{aligned}$$

Attack Process

Is it always a choice of sessions making an appropriate choice of services possible?

No:

$$
\begin{tikzcd}
M_1 \arrow[r, "\alpha^{r_1}"] \arrow[d, Rightarrow] & M_2 \arrow[d, Rightarrow] \\
\bullet & \bullet \arrow[l, "\alpha^{r_2 K_{12}}"']
\end{tikzcd}
$$

- Attacking M_1 requires a pair $(\alpha^x, \alpha^{x r_1 K_{12}^{-1}})$
- Obtaining $r_1 K_{12}^{-1}$ requires to use the r_1-service and
- a service containing K_{12} but all of them contain a random value uniquely originating which we cannot cancel

Use of Services

Is it always a choice of sessions making an appropriate choice of services possible?

Yes, for protocols involving at least 3 participants!

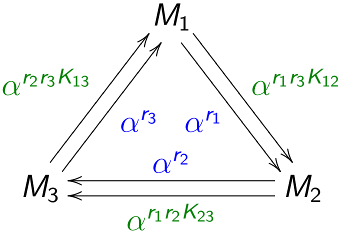
Interesting points:

- We need protocol involving at least 3 group members
- At most 3 sessions are to be considered
- Several ways of writing secrets as product of services
- It is possible for all group members

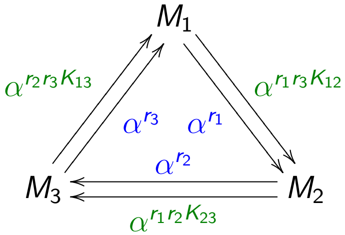
Combining services

Is this sufficient to say that all protocols of the family we consider are insecure?

No: The Tri-GDH Protocol



Combining services



- Attacking $M_1 \Rightarrow$ Obtaining a pair $(\alpha^x, \alpha^{x r_1 K_{13}^{-1}})$
- $K_{13}^{-1} \Rightarrow (\alpha^{r_x K_{13}}, \alpha) \Rightarrow r_x \Rightarrow (\alpha^{r_x K_{13}}, \alpha^{r_x})$
- $r_1 \Rightarrow$
  1. r_1? No: both r_1 and r_x have fixed inputs
  2. $r_1 K_{12}$? No: $(\alpha^{r_x K_{13}}, \alpha^{r_x r_1 K_{12}}) \Rightarrow r_y K_{12} \rightarrow$ $(\alpha^{r_x K_{13} r_y K_{12}}, \alpha^{r_x r_1 K_{12}}) \Rightarrow r_y$ but both r_x and r_y have fixed inputs

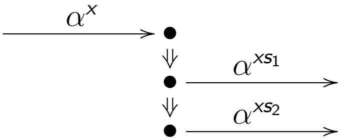
Combining services

First type of problematic services:

- *Starting Services*, i.e. services with input fixed to α

Second type of problematic services:

- *Splitting Services*, i.e. if we need to use different services with same inputs



- We can only obtain $(\alpha^{xs_1}, \alpha^{xs_2})$ (or $(\alpha^{xs_2}, \alpha^{xs_1})$)

Combining services

We defined a number of sufficient conditions making the collection of the required services possible

- The services we must collect may involve one pair of splitting services but no starting service
- The services we must collect may involve one starting service for each term of pair, but no splitting services ($\approx$)
- ...

We checked that at least one of these conditions is verified for any Cliques-type GDH-Protocol with at least 4 participants

Conclusion

We can systematically break any Cliques-type AGKAP with at least four parties.

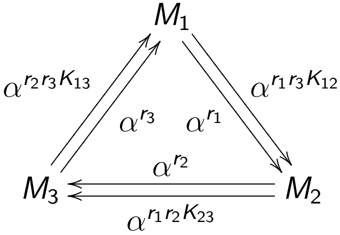
1. Use our expression of secrets as product of services and select an appropriate set of services verifying one of our sufficient conditions on splitting and starting services
2. Collect the required services for obtaining the pair $(\alpha^{x}, \alpha^{xs_i})$
3. Submit α^{x} as the value M_i will use to compute his view of the group key

- We need to consider at most three protocol sessions
- With n parties, the attacker needs to interact with at most $n+1$ strands

Open Questions

Tri-GDH Protocol:

- What could computational crypto say about this protocol?
- Could an assumption such as Pseudo-freeness help?



Open Questions

$\alpha^{xy}, \{\!|\alpha^{y}|\!\}_{K_{AB}}$?

- Cliques-type protocols with MAC's, signature, encryption, products, ...
- Addressed [Shmatikov & al. 03-04, Boreale & al. 03, Chevalier & al. 03, Kapur & al. 03, ...]

- Transpose our impossibility result to other classes of protocols?
- Proving other protocols secure when considering an infinite number of sessions?