A Security Analysis of the Cliques Protocols Suites

Olivier Pereira*- Jean-Jacques Quisquater

UCL Crypto Group,
Place du Levant, 3, B-1348 Louvain-la-Neuve,Belgium.
E-mail: {pereira,quisquater}@dice.ucl.ac.be

Abstract

Secure group protocols are not easy to design: this paper will show new attacks found against a protocol suite for sharing key. The method we propose to analyse these protocols is very systematic, and can be applied to numerous protocols of this type. The A-GDH.2 protocols suite analysed throughout this paper is part of the Cliques suites that propose extensions of the Diffie-Hellman key exchange protocol to a group setting. The A-GDH.2 main protocol is intended to allow a group to share an authenticated key while the other protocols of the suite allow to perform dynamic changes in the group constitution (adding and deleting members, fusion of groups, ...). We are proposing an original method to analyse these protocols and are presenting a number of unpublished flaws with respect to each of the main security properties claimed in protocol definition (key authentication, perfect forward secrecy, resistance to known-keys attacks). Most of these flaws arise from the fact that using a group setting does not allow to reason about security properties in the same way as when only two (or three) parties are concerned. Our method has been easily applied on other Cliques protocols and allowed us to pinpoint similar flaws.

1. Introduction

The experience has shown that the design of security protocols that are proof against active attackers is a particularly hard task. "Cryptographic" approaches have been proposed to analyse these protocols: we can notably mention the work of Bellare and Rogaway [3] that has been extended by Blake-Wilson, Johnson and Menezes [4] in order to enable the handling of the authenticated Diffie-Hellman key exchange (that we will be studying here). In these approaches, cryptographic operations are considered as functions on string of bits and security properties are expressed in terms of the probability and computational complexity of successful attacks. Proofs through this type of methods are unfortunately often laborious and do not make needless analysis methods reasoning at a higher level of abstraction. On this level, security properties are formally (logically) modelled and cryptographic operations are viewed as functions on a space of symbolic expressions. These analyses allowed researchers to capture a lot of useful intuitions about security protocols and to discover many new flaws by considering only idealized cryptographic primitives and without precisely taking into account the computational capabilities of the intruder (a first attempt to bridge the gap between these to views of cryptography can be found in [1] for example). The method we are presenting in this paper can be placed in the second category although it captures arithmetic properties from a level of abstraction lower than the one usually considered in this type of methods.

The protocols suite we will be studying here is the A-GDH.2 suite that has been proposed within the scope of the Cliques project (see [2] for instance). The main A-GDH.2 protocol (that will be referenced as the A-GDH.2 protocol in the rest of this paper) allows a group of users to agree on a contributively generated key. The other protocols of the suite permit the addition of new members in the group (A-GDH.2-MA), the removal of a member, the fusion of two groups, etc.

The analysis of these protocols raises a number of problems that have not (or not much) been studied in the literature: taking into account low-level arithmetic properties, variable number of participants in the protocols, re-use of values in several protocols, ... Furthermore, the intended security properties are not simple transpositions of those studied in the context of two parties protocols.

In this paper, we are proposing a simple model that we will use to reason about the A-GDH.2 protocol suite. The analysis we will perform with this model will lead us to pinpoint several attacks against these protocols. These attacks are typically performed by the intruder using the computations performed by honest users to obtain some secrets at the cost of the exclusion of a member from the group (which

*This author would like to thank the Belgian Funds for Scientific Research (FNRS) for its financial support.

is computing a corrupted key or not receiving some message). This exclusion, that would be very problematic in the case of two-parties protocols, has many chances to remain unnoticed by the other members of the group, particularly when the group size increases. It can also be interpreted as a network problem or as a temporary absence, which will not prevent the other members to use the key they computed.

This paper is organized as follows. First we will briefly define the A-GDH.2 protocols. Then we will explain the main particularities they present with respect to the usually analysed protocols and propose a model that we will use to perform our analysis. This analysis will constitute the last part of the text.

2. The A-GDH.2 Protocol Suite

All protocols proposed within the scope of the CLIQUES project are based on the difficulty of a single problem: the Diffie-Hellman decision (DDH) problem (i.e. given a large integer p and knowing $\alpha^a \bmod p$ and $\alpha^b \bmod p$, it is difficult to compute $\alpha^{ab} \bmod p$). All arithmetic throughout this paper will be performed in a cyclic group G of prime order q which is typically (but not necessarily) a subgroup of $\mathbb{Z}_p^*$ for a prime p such that $p = kq + 1$ for small $k \in \mathbb{N}$ (e.g. $k = 2$). We assume that p, q and α are public and known by all users, and that every user M_i shares (or is able to share) with each M_j a distinct secret K_{ij}. For example, we can set $K_{ij} = F(\alpha^{x_i x_j} \bmod p)$ where x_i is a secret long-term exponent selected by every M_i and $\alpha^{x_i} \bmod p$ is the corresponding long-term public key. We will now describe the two protocols studied in this paper: the Key Generation and the Member Adding protocols (the other protocols of the suite are not described in detail in the literature).

2.1. The A-GDH.2 Protocol

Let $M = \{M_1, \dots, M_n\}$ be a set of users wishing to share a key S_n. The A-GDH.2 protocol executes in n rounds. In the first stage ($n - 1$ rounds), contributions are collected from individual group members and then, in the second stage (n-th round), the group keying material is broadcast. The actual protocol is as follows:

Initialization:
Let p be a prime integer and q a prime divisor of $p - 1$. Let G be the unique cyclic subgroup of $\mathbb{Z}_p^*$ of order q, and let α be a generator of G.

Round i $(0 < i < n)$:

1. M_i selects $r_i \in \mathbb{Z}_q^*$
2. $M_i \to M_{i+1} : \{\alpha^{\frac{r_1 \dots r_i}{r_j}} | j \in [1, i]\}, \alpha^{r_1 \dots r_i}$

Round n:

1. M_n selects $r_n \in \mathbb{Z}_q^*$
2. $M_n \to$ All M_i: $\{\alpha^{\frac{r_1 \dots r_n}{r_i}.K_{in}} | i \in [1, n[\}$

Upon receipt of the above, every M_i computes the group key as:

$$S_n \;=\; \alpha^{(\frac{r_1 \dots r_n}{r_i}.K_{in}).K_{in}^{-1}.r_i} = \alpha^{r_1 \dots r_n}$$

The main security properties that this protocol is intended to provide are the following:

- Implicit Key Authentication: each $M_i \in M$ is assured that no party $M_q \notin M$ can learn the key $S_n(M_i)$ (i.e. M_i's view of the key) unless helped by a dishonest $M_j \in M$.
- Perfect Forward Secrecy: the compromise of long-term key(s) cannot result in the one of past session keys.
- Resistance to Known-Keys Attacks: the compromise of a session key cannot result in a passive adversary to compromise keys of other sessions, nor in an active adversary to impersonate one of protocol's parties.

All these properties have to be fulfilled in the presence of an active adversary who can insert, delay or delete messages.

2.2. The A-GDH.2-MA Protocol

Let $M = M_1, \dots, M_n$ be a set of users sharing a key S_n and assume that M_{n+1} is wishing to join the group. The A-GDH.2-MA protocol executes in 2 rounds: in the first one, M_n sends to M_{n+1} a message computed from the one he broadcast in the last round of the A-DGH.2 protocol and from the old key while in the second round, M_{n+1} broadcast the new keying material to the group. The actual protocol is as follows:

Round 1:

1. M_n selects $\hat{r}_n \in \mathbb{Z}_q^*$
2. $M_n \to M_{n+1} : \{\alpha^{\hat{r}_n \frac{r_1 \dots r_n}{r_i} K_{in}} | i \in [1, n]\}, \alpha^{\hat{r}_n r_1 \dots r_n}$

Round 2:

1. M_{n+1} selects $r_{n+1} \in \mathbb{Z}_q^*$
2. $M_{n+1} \to$ All M_i: $\{\alpha^{\hat{r}_n \frac{r_1 \dots r_{n+1}}{r_i}.K_{in}.K_{in+1}} | i \in [1, n+1]\}$

Upon receipt of the above, every M_i computes the new group key as:

$$\begin{aligned} S_{n+1} &= \alpha^{(\hat{r}_n \frac{r_1 \dots r_{n+1}}{r_i}.K_{in}.K_{in+1}).K_{in}^{-1}.K_{in+1}^{-1}.r_i} \\ &= \alpha^{\hat{r}_n r_1 \dots r_{n+1}} \end{aligned}$$

The security properties described for the A-GDH.2 protocol are intended to be preserved after the execution of the A-GDH2.MA protocol.

3. A Model for the Analysis of the Cliques Protocols

A number of methods were developed during the last few years for the “formal” analysis of security protocols. Many of them are based on state-space exploration: they usually proceed by defining an arbitrarily bounded system and explore it hoping that if there is an error in the protocol, it can be described by a behavior included in the considered state-space ([6], [7], [8], …). However several tools allow to obtain proofs for unbounded systems at the cost of the interactive proof of several lemmas [9] or of the risk of receiving no answer for some protocols [13]. Other approaches are based on the use of logics ([12], [14], …). They allow to obtain proofs for arbitrary size systems, but they often require particularly error-prone formalization steps and do not provide the same support in pinpointing problems as the direct generation of counter examples. Recently, “manual” approaches were presented, allowing to obtain fine-grained proofs for systems of any dimension, and even to analyse the interactions between protocols that can be executed concurrently (see [16] for example). In order to make such proofs feasible, several simplifying assumptions are typically stated: a very limited set of cryptographic primitives is considered (typically public-key and symmetric-key encryption), and these primitives are usually idealized in such a way that they act as black-boxes (ignoring low-level properties such as the multiplicative structure of RSA or the characteristics of the chaining method used in symmetric-key encryption for example). The use of state-space exploration techniques in the study of group-protocols seems very difficult due to their very essence: the number of participants in a honest session of the protocol is basically unbounded, what will intuitively result in dramatic state-space explosion problems. As we know, the only successful analyses of group protocols have been performed by theorem proving approaches ([5], [11]), which allow inductive reasoning. However we recently learned that C. Meadows was performing (independently of us) the analysis of the A-GDH.2 protocol, adapting her NRL Protocol analyzer by extending the power and scope of its theorem-proving capabilities [10]. Beyond the problem of the unbounded number of participants in the protocols, the modelling of the A-GDH.2 protocols suite requires the capturing of several low-level arithmetic properties: exponentiation, commutativity, associativity, that are out of the scope of most of the works encountered in the literature. Furthermore, the A-GDH.2 key generation protocol is not intended to be used alone: there are several other protocols in the suite (member addition, …) that use values computed during the key generation protocol and can interfere with its security properties. All these characteristics led us to adapt ideas presented in the context of the strand space approach ([15], …) in order to be able to reason about protocols based on the Diffie-Hellman Decision problem. In the following paragraphs, we will first introduce the modeling of the messages that we are using, then we will describe the intruder capabilities and, finally, we will show how the intended security properties can be verified and apply our method for the analysis of the A-GDH.2 protocols.

3.1. Messages and Intruder’s Knowledge

The messages sent in the protocols proposed within the scope of the CLIQUES project are constituted by the concatenation of elements of a group G of prime order q. A particular element, that we will denote α, is a generator of G and is shared by all users of the network (as well as the knowledge of the characteristics of the group G). All exchanged elements of G are expressed as powers of $\alpha \bmod p$. It can then be checked that the participants have to manipulate three types of elements:

- Random Numbers (r_i)
- Long-term Keys (K_{ij})
- Elements of G expressed as α raised to the power of a product of random numbers and long-term keys. We will denote the set of all these product as P (i.e. $P = \{\prod r_i^{e_i} \prod K_{jl}^{e_{jl}} | e_i, e_{jl} \in \mathbb{Z}\}$. The only sent elements are the ones of this type.

The behavior of the honest participants is quite simple: they receive elements of G, exponentiate them with random numbers and/or long-term key (possibly inverted), and send them to other participants. The group-key is obtained in the same manner, except that the result of the computations is not sent but kept secret. It can be noticed that when a participant receives an element of G, he has to accept it without being able to check anything concerning its constitution or origin. Furthermore, in the key-generation protocol described above, the completion of a protocol session does not implies for any M_i the aliveness of an other expected group member: the expected implicit key-authentication property says that the key computed by M_i at the reception of the broadcast of the n-th round of the protocol (key that we will write $S_n(M_i)$) can be known only by the participants to whom this message was broadcast by M_n (if M_n actually sent this message). The goal of an intruder will therefore be to possess a pair of elements of G related between them in such a way that the second is equal to the result of the key-computation operations of an honest M_i applied to the first element of the pair. If we take this point of view, there are n secret pairs corresponding to an execution of the protocol between n parties. As we said above, the key-computation operation is always a sequence of exponentiations of a received element of G by some previously generated random numbers or keys. In a scriptural view, these operations amount to multiply an element of P by another (secret) element of P and to keep the result confidential. We can then define a set R as the set of the ratios between elements of P, and the goal of the intruder will be to obtain some secret value of R.

More precisely, our model will deal with two sets of elements:

- The set E containing the random numbers r_i and the long-term keys K_{ij}
- The set R of the ratios between elements of P. This set is defined as follows: given the set E and an injective function $f : E \to R$, $(R, .)$ is the commutative group of which the elements of $\mathcal{IM}(f)$ (the image of E in R trough the f-function) are the generators. In order to simplify the notations, we will use the same letter to denote $e \in E$ and $f(e) \in R$.

Example: After receiving the broadcast from M_n, M_1 extracts the first element of this message (that he hopes to be $\alpha^{r_2,\ldots,r_n K_{1n}}$) and exponentiates it with $r_1 K_{1n}^{-1}$ to obtain its view of the group key. The "challenge" for the intruder is hence to find a pair of elements of G of the type $(\alpha^x, \alpha^{x r_1 K_{1n}^{-1}})$, so he can send α^x to M_1 through M_n's broadcast and know the key computed by M_1. With our notations, the secret pair corresponding to the secret of M_1 will be represented by the element $\frac{x r_1 K_{1n}^{-1}}{x} = r_1 K_{1n}^{-1} \in R$.

The use of such a construction will be quite convenient and can be intuitively justified if we state a hypothesis that is quite similar to the widely used "perfect encryption assumption". We will refer to this hypothesis as the "perfect Diffie-Hellman assumption" and it can be stated as follows:

> "An element of G can be computed in one and only way: by exponentiating the generator α with the correct random numbers and keys (excepted the permutations in the order of the exponentiation of α and the possibility of exponentiation by an element of E and by its invert successively)."

This assumption implies in particular that a secret cannot be computed by combining elements of G (but only elements of E with elements of G). It seems quite plausible in the practice given that we work within a large group (lucky guesses or collisions are very unlikely) and that the DDH problem is hard.

It can also be noticed that the use of the R-set implies another restriction due to its very structure: it does not allow capturing relations between more than two elements of G. Once again, it does not seem to be a problem if we notice that the relevant security properties always come down to the impossibility of finding two elements of G presenting between them a particular relation, so that the consideration of more complex relations cannot be of any help to prove the correctness of Cliques protocols. It could be useful to use such extensions to discover more dangerous attacks that violate more than one security property, but we are more interested in checking the correctness of protocols than in finding "optimal" attack sketches. We will now be looking at the ways that the intruder can use to manipulate our two sets of elements.

3.2. Intruder Capabilities

Considering our "perfect Diffie-Hellman assumption", the only useful computation for the intruder will be the exponentiation of an element of G by a known element of E. If we note E_I and R_I the subsets of elements of E and R (respectively) that are known by the intruder, we can then transpose this remark as follows:

$$\text{(1) If } e \in E_I \text{ and } r \in R_I \text{ then} \\ r.e \in R_I \text{ and } r.e^{-1} \in R_I$$

There is another way for the intruder to obtain new elements of R: the use of the computations executed by the honest users. As we said above, the behavior of these users is quite simple: they receive elements of G and exponentiate them with some values of E. We will call such operations *services*. More precisely, a service is a function $s : G \to G : \alpha^x \to \alpha^{p \cdot x}(x, p \in P)$, and we call S the set of the available services. Let us see how a service can be described in term of growth of R_I. If $r \in R_I$, then the intruder possesses two elements of G that can be written α^x and α^{rx}. If the intruder sends α^x to an honest user performing the service $s : s(\alpha^x) = \alpha^{px}$, then he will learn the element $p^{-1}.r \in R_I$. Conversely, if the intruder sends α^{rx} to the user that performs the same service, he will learn the element $p.r \in R_I$. We can then write our second rule for the increasing of the R_I-set:

$$\text{(2) If } s \in S : s(\alpha^x) = \alpha^{p \cdot x}, \text{ and } r \in R_I \text{ then} \\ r.p \in R_I \text{ or } r.p^{-1} \in R_I$$

Nevertheless we have to be careful in the use of this rule and impose some restrictions in its application due to the fact that the honest users provide several services in parallel and only once. This will be examined more in the detail in the next section where we will propose a method to determine if a ratio is secret or can be obtained by the intruder.

3.3. Proving Security Properties

In the context of the Cliques protocols, the most general message transformation provided by a user during a single round can be written as follows:

$\alpha^{x_1}.\alpha^{x_2}\dots\alpha^{x_n} \rightarrow$

$$\alpha^{x_1 y_{11}}.\alpha^{x_1 y_{12}}\dots\alpha^{x_2 y_{21}}.\alpha^{x_2 y_{22}}\dots\alpha^{x_n y_{n1}}\dots\alpha^{x_n y_{nm}}$$

This view can be used to express the rules limiting the composition of services in the derivation of the set R_I:

- The rule (2) can be used at most once for each service. Furthermore, it can only be used on an element of R_I that has been obtained previously.

- If two services $s_1 : s_1(\alpha^x) = \alpha^{p_1 x}$ and $s_2 : s_2(\alpha^x) = \alpha^{p_2 x}$ are performed during the same round and take distinct inputs (i.e. are applied to distinct α^{x_i}), then they can be used on a single element $r \in R_I$ to produce the following elements: $r.p_1$, $r.p_2$, $r.p_1^{-1}$, $r.p_2^{-1}$, $r.p_1^{-1}.p_2$, $r.p_1.p_2^{-1}$ (but not $r.p_1.p_2$ nor $r.p_1^{-1}.p_2^{-1}$)

- If two services $s_1 : s_1(\alpha^x) = \alpha^{p_1 x}$ and $s_2 : s_2(\alpha^x) = \alpha^{p_2 x}$ are performed during the same round and take the same inputs (i.e. are applied to the same α^{x_i}), then they can be used to produce the following elements: $p_1^{-1}.p_2$ or $p_1.p_2^{-1}$. It can be noticed that these elements are independent from any previously known element of R_I.

From these considerations, we can suggest a systematic scheme to obtain the proof of the secrecy of a particular $r \in R$.

1. Expression of the available services (S-set), of the atomic elements and ratios initially known by the intruder (E_I and R_I), and of the secret ratios (let R_S be this set).

2. Deletion of all elements corresponding to those of E_I from the expression of S, R_I, and R_S. This operation simplifies the problem and does not change its solutions since:

   - If $e \in E_I$, every operation that uses the service $s : s(\alpha^x) \rightarrow \alpha^{xe_i y} (i \in \mathbb{Z})$ can be performed by using a service $s' : s'(\alpha^x) \rightarrow \alpha^{xy}$ and by suitably applying the (1)-rule.

   - If $e \in E_I$, and $r.e^a \in R_I$ then $r \in R_I$ (anew by applying rule (1))

   - If $e \in E_I$, and $r.e^a \in R_S$ then the knowledge of r implies the one of $r.e^a$ (for the same reason)

   Example: if $K_{1n} \in E_I$ then the service $s : s(\alpha^x) \rightarrow \alpha^{xr_n K_{1n}}$ is as useful as the service $s' : s'(\alpha^x) \rightarrow \alpha^{xr_n}$.

3. Writing of the linear system expressing the "balance" of the variables in the construction of the secret from the services. This system contains one variable per service and element of R_I, one equation per element in E, and the second term of each equation is the power of the corresponding element in the studied secret. This system expresses that the only way to compute the secret is to successively apply some services on a known ratio. If this system is inconsistent, then the intended confidentiality property is verified (in our model). If this is not the case, we have to check the restrictions on the use of services described above. If it is not possible to find a solution of the system that meets all these constraints, then no attack on the protocol can be derived from our model (else an unwanted scenario can be derived from the list of services that have to be used).

We will now see how this scheme can be applied for the analysis of the A-GDH.2 protocols suite.

4. Analysis of the A-GDH.2 Protocols

In the first paragraphs, we will concentrate our study on the three security properties of the A-GDH.2 key generation protocol described in section 2.1. Then, we will extend our analysis by considering the concurrent use of the A-GDH.2-MA protocol.

4.1. Implicit Key Authentication

As described above, the first step in our analysis will be the description of the protocol.

Initially, we will consider only one session of the protocol. The intruder knowing no long-term keys nor short-term secrets, the E_I is be empty. In the first round, the user M_1 provides $r_1 \in R_I$. From the second round to the $(n-1)$-th round, the user M_i provides the service $s_i : s_i(\alpha^x) \rightarrow \alpha^{xr_i}$ several times in parallel. For the simplicity, we will refer to the service $s_i(\alpha^x) \rightarrow \alpha^{xr_i}$ by the power it raises its input: $r_i (\in S)$. During the n-th round, M_n provides the $n-1$ services: $r_n K_{1n}, \dots, r_n K_{n-1n}$. The secrets are the following: $r_i K_{in}^{-1}$ for M_i $(1 \leq i < n)$ and r_n for M_n. So, to summarize, the sets we will consider are the following:

$$\begin{array}{rcl} S & = & \{r_2, \dots, r_{n-1}, r_n K_{1n}, \dots, r_n K_{n-1n}\} \\ E_I & = & \emptyset \\ R_I & = & \{r_1\} \\ R_S & = & \{r_1 K_{1n}^{-1}, \dots, r_{n-1} K_{n-1n}^{-1}, r_n\} \end{array}$$

If we follow the analysis scheme proposed above, we have now to express the linear system describing the "balance" of the variables of E to check the secrecy of the elements of R_S. We will first look at the secrecy of $r_1 K_{1n}^{-1}$ (so we need the balance on r_1 and K_{1n} being equal to 1 and -1 respectively while the balance on the other variables must be null). If we use the "s"-letter to denote the coefficient of the variable indicating how many times the service s has to be used to construct the secret, it can be written as follows:

$r_1 = 1$	Balance on r_1
$r_i = 0$	Balance on r_i $(2 \leq i < n)$
$\sum_{i=1}^{n-1} r_n K_{in} = 0$	Balance on r_n
$r_n K_{1n} = -1$	Balance on K_{1n}
$r_n K_{in} = 0$	Balance on K_{in} $(2 \leq i < n)$

It can be observed that the summing of the $n-1$ equations corresponding to the balance on the keys provides an inconsistency with the equation expressing the balance on r_n. Hence we can say that $r_1 K_{1n}^{-1}$ cannot be obtained by using the two enrichment rules we defined and that $S_n(M_1)$ is kept secret in our model as claimed in protocol definition. If we write this system in the case of multiple sessions of the protocol (from which the intruder is excluded), it can be easily checked that this inconsistency is preserved. The transposition of this result for the $r_i K_{in}^{-1}$ -secrets is straightforward and if we transform the second members of these equations in order to prove the secrecy of r_n, we can easily obtain an inconsistency between the same equations. We can then say that the Implicit Key Authentication property is correct with respect to our model provided that the intruder is not a member of any group.

We will now check if this property is preserved when the intruder is a member of some groups. As a simple scenario, we will assume a first session of the protocol in which M_1, $\ldots$, M_n and I are the intended participants (and M_n is the group controller), and a second of which I is excluded. We will note r'_i the random integer generated by M_i during the second session. The sets of interest hence become:

$$\begin{aligned} S &= \{r_2, \ldots, r_{n-1}, r_n K_{1n}, \ldots, r_n K_{n-1n}, r_n K_{In}, \\ &\quad\; r'_2, \ldots, r'_{n-1}, r'_n K_{1n}, \ldots, r'_n K_{n-1n}\} \\ E_I &= \{K_{In}\} \\ R_I &= \{r_1, r'_1\} \\ R_S &= \{r'_1 K_{1n}^{-1}, \ldots, r'_{n-1} K_{n-1n}^{-1}, r'_n\} \end{aligned}$$

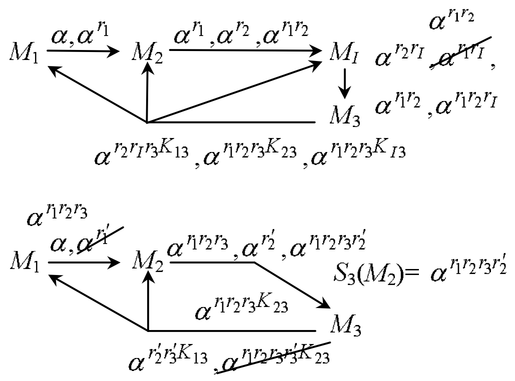
If we solve the corresponding linear systems, a number of solutions can be found. Among them, we can check that the secret $r'_i K_{in}^{-1}$ can be obtained by using the services r'_i and $r_n K_{In}$ in the positive direction, by using the service $r_n K_{in}$ in the inverted direction, and by taking into account the knowledge of K_{In} by the intruder (several other solutions of the same system are usable). The corresponding scenario is as follows:

- During the next to last round of the first session of the protocol, I replaces $\alpha^{r_1 \cdots r_{i-1} r_{i+1} \cdots r_{n-1} r_I}$ by $\alpha^{r_1 \cdots r_i \cdots r_{n-1}}$. Hence, M_n will send $\alpha^{r_1 \cdots r_i \cdots r_n K_{in}}$ and $\alpha^{r_1 \cdots r_i \cdots r_n K_{In}}$. I will compute the key shared by every $M_j (1 \leq j \leq n)$ but M_i, and M_i will compute a wrong key.

- I replaces one of the inputs of the i-th round of the second session of the protocol with $\alpha^{r_1 \cdots r_i \cdots r_n}$ (computed in the previous step), and M_i then sends $\alpha^{r_1 \cdots r_i \cdots r_n r'_i}$ to M_{i+1}.

- In the broadcast of the last round of the protocol second session, I replaces the contribution intended to M_i with $\alpha^{r_1 \cdots r_i \cdots r_n K_{in}}$. The secret value computed by M_i will thus be $\alpha^{r_1 \cdots r_i \cdots r_n r'_i}$ that the intruder learned during the previous step of this scenario.

An instantiation of this scenario is represented for $n = 3$ and $i = 2$ in Fig. 1.



M_1
M_2
M_I
M_3
α, α^{r_1}
$\alpha^{r_1}, \alpha^{r_2}, \alpha^{r_1 r_2}$
$\alpha^{r_1 r_2}$
$\alpha^{r_2 r_I}$ ~~$, \alpha^{r_1 r_I}$~~ ,
$\alpha^{r_1 r_2}, \alpha^{r_1 r_2 r_I}$
$\alpha^{r_2 r_I r_3 K_{13}}, \alpha^{r_1 r_2 r_3 K_{23}}, \alpha^{r_1 r_2 r_3 K_{I3}}$
$\alpha^{r_1 r_2 r_3}$
$\alpha,$ ~~$\alpha^{r'_1}$~~
$\alpha^{r_1 r_2 r_3}, \alpha^{r'_2}, \alpha^{r_1 r_2 r_3 r'_2}$
$S_3(M_2) = \alpha^{r_1 r_2 r_3 r'_2}$
$\alpha^{r_1 r_2 r_3 K_{23}}$
$\alpha^{r'_2 r'_3 K_{13}}$ ~~$, \alpha^{r_1 r_2 r_3 r'_3 K_{23}}$~~

Figure 1. Attack against Implicit Key Authentication

It can be observed that a similar scenario can be applied in parallel against all members of the first group, and thus that the intruder is able to share a (different) key simultaneously with all the members of the second group (from which he is normally excluded). Finally, the implicit key authentication property seems to be very problematic in this protocol as soon as the intruder is a member of a group and intended to be excluded from another non disjoint group.

We will now analyse the other security properties, considering that I is not a member of any group.

4.2. Perfect Forward Secrecy

In the study of this property, we will assume that E_I contains all long-term keys K_{in}. If we consider only one session of the protocol and apply the transformation suggested as second step of our proof-scheme, we can rewrite the set of services $S = \{r_i | i \in [2, n]\}$, $R_I = \{r_1\}$, and R_S as $\{r_i | (i \in [1, n]\}$. For each secret r_i, the resulting linear system has a trivial solution: $r_i = 1$. These solutions meet all restrictions described above, and we can then assume that the perfect forward secrecy is somehow suspicious.

A scenario corresponding to an attack against M_1 is as follows. The secret is $r_1 K_{in}^{-1}$ and the value of interest is r_1 provided by M_1 in the first round. The intruder will therefore replace the element of G intended to M_1 in the broadcast of the n-th round by α in such a way that $S_n(M_1)$ will be computed as $\alpha^{r_1 K_{1n}^{-1}}$. The perfect forward secrecy property says that the compromising of long-term keys cannot result in the one of session keys. But if K_{1n} is compromised, the intruder will be able to compute $\alpha^{r_1 K_{1n}^{-1}}$ (by exponentiating α^{r_1} provided during the first round). Hopefully, this problem does not seem very dangerous in the practice since $S_n(M_1)$ will be different of the keys computed by the other members of the group. The scenario will be similar for all the other $M_i (i < n)$, and it can be noticed that all these attacks can be performed in parallel, which can be useful in some contexts. However the attack against M_n will be somewhat different. His secret is r_n, and the useful services are $r_n K_{1n}$, …, $r_n K_{n-1n}$ (each can be used). These services are respectively applied to the $n - 1$ first elements of the $(n - 1)$-th round, and the secret is computed from the last element of the same run. The intruder will then proceed by substituting one of the $n - 1$ first elements of this round with the last element of the message. If we suppose that he substitutes the first element, M_n will compute $S_n(M_n) = \alpha^{r_1 \cdots r_n}$ and broadcast $\alpha^{r_1 \cdots r_n K_{1n}}$, $\alpha^{r_1 r_3 \cdots r_n K_{2n}}$, …, $\alpha^{r_1 r_2 \cdots r_{n-1} r_n K_{n-1n}}$. Hence M_1 will be computing a wrong key: $S_n(M_1) = \alpha^{r_1^2 \cdots r_n}$ while all other members of the group will compute $S_n(M_i) = \alpha^{r_1 \cdots r_n} (1 < i \leq n)$.

Then, if K_{1n} is compromised, the intruder will be able to compute the key $S_n(M_n)$ that is shared by all group-members except one (which he can isolate from the rest of the network or that can have never been alive). This attack is represented for a group of four members in Fig. 2. It seems to us that this is a much more awkward scenario.

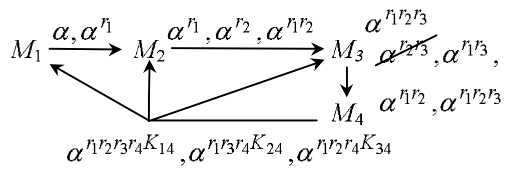

M_1
α, α^{r_1}
M_2
$\alpha^{r_1}, \alpha^{r_2}, \alpha^{r_1 r_2}$
M_3
$\alpha^{r_1 r_2 r_3}$
$\cancel{\alpha^{r_2 r_3}}, \alpha^{r_1 r_3},$
M_4
$\alpha^{r_1 r_2}, \alpha^{r_1 r_2 r_3}$
$\alpha^{r_1 r_2 r_3 r_4 K_{14}}, \alpha^{r_1 r_3 r_4 K_{24}}, \alpha^{r_1 r_2 r_4 K_{34}}$

Figure 2. Attack against Perfect Forward Secrecy

The fact that this attack provides the key computed by group-members others than M_n corresponds to the fact that it exploits solutions of the type $r_i = 1$, $r_n K_{in} = -1$, $r_n K_{jn}$ =1 that are less trivial solutions of the system corresponding to the secret of M_i.

4.3. Resistance to known-keys attacks

This property expresses that the compromising of session keys does not allow a passive adversary to compromise keys of other sessions nor an active adversary to impersonate one of the protocol parties. The part of this property concerning the passive adversary is studied in [2] and we will focus on the second part. However the authors claim that the resistance to an active adversary is more dubious and suggest an attack that does not seem very useful in the practice. The application of our method to the verification of this property is as follows. We will assume two sessions of the protocol with the same participants, and the random numbers generated by M_i during the first and second sessions of the protocol will be r_i and r'_i respectively. Hence we can write that

$$\begin{aligned} S &= \{r_2, \ldots, r_{n-1}, r_n K_{1n}, \ldots, r_n K_{n-1n}, \\ &\quad r'_2, \ldots, r'_{n-1}, r'_n K_{1n}, \ldots, r'_n K_{n-1n}\} \\ E_I &= \emptyset \\ R_I &= \{r_1, r'_1, r_1 K_{1n}^{-1}, \ldots, r_{n-1} K_{n-1n}^{-1}, r_n\} \\ R_S &= \{r'_1 K_{1n}^{-1}, \ldots, r'_{n-1} K_{n-1n}^{-1}, r'_n\} \end{aligned}$$

If we write the linear system corresponding to $r'_i K_{in}^{-1} (1 \leq i < n)$, we can check that

$$r_i K_{in}^{-1} = 1, r_i = -1, r'_i = 1$$

and all other services unused is a solution. If $i = 1$, it is however impossible to find an attack scheme since all these values are in R_I and cannot be successfully assembled. Nevertheless, for all others values of i, the following attack is possible:

1. Let α^x be one of the terms of the input of the i-th round of the first run of the protocol. M_i will therefore send $\alpha^{x r_i}$.

2. The intruder replaces then the term $\alpha^{r_1 \cdots r_{i-1} r_{i+1} \cdots r_n K_{in}}$ with α^x. Hence $S_n(M_i)$ will be equal to $\alpha^{x r_i K_{in}^{-1}}$. Since we study known-keys attacks, we will assume that this value is compromised.

3. In the second run of the protocol, the intruder replaces one of the i-th round inputs with $\alpha^{x r_i K_{in}^{-1}}$. M_i will therefore send $\alpha^{x r_i K_{in}^{-1} r'_i}$.

4. In the broadcast of the second run of the protocol, the intruder finally replaces the term intended to M_i with $\alpha^{x r_i}$ (obtained in the first step of our scenario). Hence $S'_n(M_i)$ will be computed as $\alpha^{x r_i K_{in}^{-1} r'_i}$ that has been obtained during the third step of our scenario.

At the end of this scenario, the intruder will possess a key that M_i believes to be secret. However this key is unknown to the rest of the group and the compromised key used is a malformed key which reduces the scope of these attacks. However, if all malformed keys are available, the intruder can perform this attack simultaneously against almost all members of the group...

We can now turn to the secrecy of r'_n. If we look at the linear system corresponding to this secret, we can find two types of solutions. The first one is:

$$r_i = -1, r_i K_{in}^{-1} = 1, r'_n K_{in} = 1$$

From these solutions, we can obtain the scenarios corresponding to the attack proposed in [2]. The scope of these attacks is the same as the one we just described.

However another type of solution can be found:

$$r_n = 1, r_n K_{in} = -1, r'_n K_{in} = 1$$

For $1 \leq i < n$, it is possible to apply the following scenario:

1. In the inputs of the last round of the first session of the protocol, the intruder replaces $\alpha^{r_1 \cdots r_{i-1} r_{i+1} \cdots r_{n-1}}$ with $\alpha^{r_1 \cdots r_i \cdots r_{n-1}}$. Hence all elements of the broadcast will be preserved except the one intended to M_i that will be equal to $\alpha^{r_1 \cdots r_n K_{in}}$. $S_n(M_n)$ will therefore be equal to $\alpha^{r_1 \cdots r_n}$ and shared by all members of the group except M_i. In a context of known-key attacks, we will assume that this key is compromised.

2. In the inputs of the last round of the protocol second session, the intruder will substitute $\alpha^{r'_1 \cdots r'_{n-1}}$ with $\alpha^{r_1 \cdots r_n K_{in}}$ and $\alpha^{r'_1 \cdots r'_{i-1} r'_{i+1} \cdots r'_{n-1}}$ with $\alpha^{r_1 \cdots r_n}$. Hence M_n will broadcast $\alpha^{r_1 \cdots r_n K_{in} r'_n}$ and compute $S_n(M_n) = \alpha^{r_1 \cdots r_n K_{in} r'_n}$.

This scenario is more dangerous since we assume the compromising of a key that has been shared (and normally used) by all members of the group except one. However it allows to attack only M_n. Fig. 3 represents this scenario for $i = 1$ and a group of four members. Consequently the resistance to known-key attacks seems problematic in this protocol.

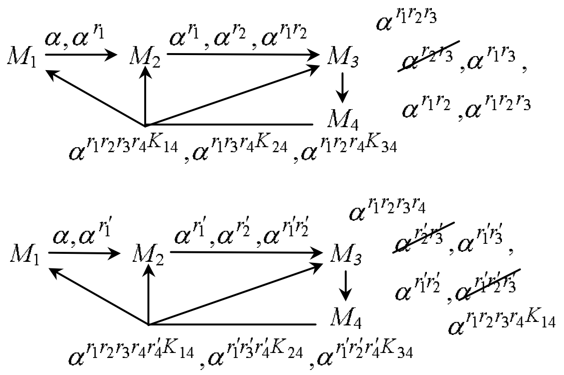

M_1
M_2
M_3
M_4
α, α^{r_1}
$\alpha^{r_1}, \alpha^{r_2}, \alpha^{r_1 r_2}$
$\alpha^{r_1 r_2 r_3}$
$\cancel{\alpha^{r_2 r_3}}, \alpha^{r_1 r_3},$
$\alpha^{r_1 r_2}, \alpha^{r_1 r_2 r_3}$
$\alpha^{r_1 r_2 r_3 r_4 K_{14}}, \alpha^{r_1 r_3 r_4 K_{24}}, \alpha^{r_1 r_2 r_4 K_{34}}$
$\alpha, \alpha^{r'_1}$
$\alpha^{r'_1}, \alpha^{r'_2}, \alpha^{r'_1 r'_2}$
$\alpha^{r_1 r_2 r_3 r_4}$
$\cancel{\alpha^{r'_2 r'_3}}, \alpha^{r'_1 r'_3},$
$\alpha^{r'_1 r'_2}, \cancel{\alpha^{r'_1 r'_2 r'_3}}$
$\alpha^{r_1 r_2 r_3 r_4 K_{14}}$
$\alpha^{r_1 r_2 r_3 r_4 r'_4 K_{14}}, \alpha^{r'_1 r'_3 r'_4 K_{24}}, \alpha^{r'_1 r'_2 r'_4 K_{34}}$

Figure 3. Attack against Resistance to Known Keys

4.4. Consideration of the Use of the A-GDH.2-MA Protocol

The key generation protocol (A-GDH.2) is not intended to be used alone: it is often useful to enable the addition or deletion of group members after the initial group creation and, in order to provide each of these services, we will use new protocols. As we said above, the aim of the A-GDH.2-MA protocol is the addition of a new member in the group. In this paragraph, we will extend our analysis of the A-GDH.2 protocol by taking into account the presence of the Member Adding protocol. As a first step, we will study the Implicit Key Authentication property and consider two sessions of the protocols: in the first session, the A-GDH.2 protocol is executed by $M_1, \ldots, M_n$; while in the second session a member is added to this group. Following the same approach as above, we will first write the sets E_I, R_I, R_S and S that will be the union of those corresponding to each of the two protocols sessions:

$$\begin{aligned}
S &= \{r_2, \ldots, r_{n-1}, r_n K_{1n}, \ldots, r_n K_{n-1n}, \\
&\quad r_n \hat{r}_n, r_n \hat{r}_n K_{1n}, \ldots, r_n \hat{r}_n K_{n-1n}, r_n \hat{r}_n K_{nn}, \\
&\quad r_{n+1} K_{1n+1}, \ldots, r_{n+1} K_{n+1n+1}\} \\
E_I &= \emptyset \\
R_I &= \{r_1\} \\
R_S &= \{r_1 K_{1n}^{-1}, \ldots, r_{n-1} K_{n-1n}^{-1}, r_n, \\
&\quad r_1 K_{1n}^{-1} K_{1n+1}^{-1}, \ldots, r_{n+1} K_{n+1n}^{-1} K_{n+1n+1}^{-1}\}
\end{aligned}$$

The first part of the expression of S corresponds to the execution of the key-generation protocol, the second to the first round of the A-GDH.2-MA protocol, and the last to the second round of the A-GDH.2-MA protocol. The two parts of the definition of R correspond to the key generation session and to the member adding protocol respectively.

E_I being empty, we can immediately study the linear system corresponding to the secrets. This system is a little larger than the previous but remains quite regular. If we solve it, we find that a number of secrets can be compromised: $r_i K_{in}^{-1} (1 \leq i < n)$ can be obtained by combining the services (or ratios in the case of r_1) r_i, $r_n \hat{r}_n$ and $r_n \hat{r}_n K_{in} (1 \leq i < n)$. The other secrets cannot be compromised in this scheme. The corresponding scenario is as follows:

1. $M_1, \ldots, M_n$ execute the key-generation protocol, but I intercepts the broadcast of the n-th round.

2. I obtains that M_n starts the A-GDH.2-MA protocol with some other user of the network, and eavesdrop the first message.

3. I sends the parts corresponding to the users $M_1, \ldots, M_{n-1}$ faking the broadcast of the A-GDH.2 protocol.

When done, $M_1, \ldots, M_{n-1}$ will share with the intruder the key $\alpha^{r_1 r_2 \cdots r_n \hat{r}_n}$ that has been sent by M_n as the last part of the first message of the A-GDH.2-MA protocol. The scheme corresponding to this attack in the case of the adding of a fourth member to the group is described in Fig. 4. Hence the Implicit Key Authentication property seems to become even more problematic when we consider the possibility of the use of the A-GDH.2-MA protocol in parallel with the A-GDH.2 protocol. The other security properties could be similarly studied but we do not think that it would be useful at this time.

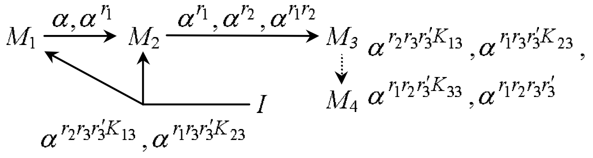

M_1
α, α^{r_1}
M_2
$\alpha^{r_1}, \alpha^{r_2}, \alpha^{r_1 r_2}$
M_3
$\alpha^{r_2 r_3 r_3' K_{13}}, \alpha^{r_1 r_3 r_3' K_{23}},$
M_4
$\alpha^{r_1 r_2 r_3' K_{33}}, \alpha^{r_1 r_2 r_3 r_3'}$
I
$\alpha^{r_2 r_3 r_3' K_{13}}, \alpha^{r_1 r_3 r_3' K_{23}}$

Figure 4. Attack considering the A-GDH.2-MA Protocol

5. Conclusion

Throughout this paper, we presented the first steps of the development of a model for the analysis of Diffie-Hellman based group key-agreement protocols and applied it on the A-GDH.2 suite of the Cliques protocols. The reasoning in our model led us to pinpoint a number of unpublished flaws in these protocols, emphasizing the necessity to be able to reason systematically on security protocols, especially in contexts where active adversaries are to be considered. We also applied our method to analyse the SA-GDH.2 protocols suite (that are designed to provide complete group authentication), and obtained similar flaws that we did not detail in this paper due to the similarity they presented with the ones we described above.

We are currently working on defining more precisely the attacks detectable (and those undetectable) within our model, on its extension in such a way that it can be used to take into account classical encryption and signature schemes, and on the construction of fixes on the A-GDH.2 protocols that are secure from our model point of view.

Acknowledgements

The authors would like to thank O. Chevassut for his useful comments on the Cliques protocols and S. Baudine for her valuable help to finish off this paper.

References

[1] M. Abadi and P. Rogaway. Reconciling two views of cryptography. In *Proceedings of the IFIP International Conference on Theoretical Computer Science 2000*, pages 3–22, 2000.

[2] G. Ateniese, M. Steiner, and G. Tsudik. New multi-party authentication services and key agreement protocols. *IEEE Journal on Selected Areas in Communication*, 2000.

[3] M. Bellare and P. Rogaway. Entity authentication and key distribution. In *Proceedings of Advances in Cryptology: Crypto'93*, pages 232–249. LNCS Vol. 773, 1994.

[4] S. Blake-Wilson, D. Johnson, and A. Menezes. Key agreement protocols and their security analysis. In *Cryptography and Coding*, pages 30–45. LNCS Vol. 1355, 1997.

[5] J. Bryans and S. Schneider. CSP, PVS, and a recursive authentication protocol. In *Proceedings of the DIMACS Workshop on Formal Verification of Security Protocols*, 1997.

[6] A. Durante, R. Focardi, and R. Gorrieri. CVS: A tool for the analysis of cryptographic protocols. In *Proceedings of the 12-th IEEE Computer Security Foundations Workshop*, pages 203–212. IEEE Computer Society Press, 1999.

[7] G. Lowe. Casper: A compiler for the analysis of security protocols. *Journal of Computer Security*, 6:53–84, 1998.

[8] W. Marrero, E. Clarke, and S. Jha. A model checker for authentication protocols. In *Proceedings of the DIMACS Workshop on Formal Verification of Security Protocols*, 1997.

[9] C. Meadows. The NRL protocol analyzer : an overview. *Journal of Logic Programming*, 26(2):113–131, 1996.

[10] C. Meadows. Extending formal cryptographic protocol analysis techniques for group protocols and low-level cryptographic primitives. In *Proceedings of the Workshop on Issues in the Theory of Security*, 2000.

[11] L. C. Paulson. Mechanised proofs for a recursive authentication protocol. In *Proceedings of the 10-th IEEE Computer Security Foundations Workshop*, pages 84–95. IEEE Computer Society Press, 1997.

[12] L. C. Paulson. The inductive approach to verifying cryptographic protocols. *Journal of Computer Security*, 6:85–128, 1998.

[13] D. Song. Athena: A new efficient automatic checker for security protocol analysis. In *Proceedings of the IEEE Symposium on Research in Security and Privacy*, 1999.

[14] P. Syverson and P. van Oorschot. On unifying some cryptographic protocols logics. In *Proceedings of the IEEE Symposium on Research in Security and Privacy*, pages 14–24, 1994.

[15] F. J. Thayer, J. H. Herzog, and J. Guttman. Mixed strand spaces. In *Proceedings of the 12-th IEEE Computer Security Foundations Workshop*, pages 83–89. IEEE Computer Society Press, 1999.

[16] F. J. Thayer, J. H. Herzog, and J. Guttman. Strand spaces: Proving security protocols correct. *Journal of Computer Security*, 7(2/3):191–230, 1999.