Security Analysis of the Cliques Protocols Suites: First results

O. Pereira, J-J. Quisquater
UCL Crypto Group

Place du Levant, 3 - B-1348 Louvain-la-Neuve, Belgium.

{pereira, quisquater}@dice.ucl.ac.be

Key words: Group Protocols, Diffie-Hellman, Systematic Analysis, Cliques Protocols.

Abstract: The Cliques protocols are extensions of the Diffie-Hellman key exchange protocol to a group setting. In this paper, we are analysing the A-GDH.2 suite that is intended to allow a group to share an authenticated key and to perform dynamic changes in the group constitution (adding and deleting members, …). We are proposing an original method to analyze these protocols and are presenting a number of unpublished flaws with respect to each of the main security properties claimed in protocol definitions (key authentication, perfect forward secrecy, resistance to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.

1. INTRODUCTION

Within the scope of the CLIQUES project, five suites of group key distribution protocols have been developed. In this paper, we are studying the A-GDH.2 protocols suite [AST00]. 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 several 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 the pinpointing of 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 is computing a corrupted key or not receiving some messages). 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 a subgroup of Z_p^* for a prime p such that $p = kq + 1$ for small $k \in \mathrm{N}$ (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 $\alpha^{x_i} \pmod p$ 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 $M = \{M_1, \ldots, M_n\}$ 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 Z_p^* of order q, and let α be a generator of G.
Round i ($0 < i <$ n)**:**

1. M_i selects $r_i \in Z_q^*$
2. $M_i \rightarrow M_{i+1}$: $\{ \alpha^{\frac{r_1 \ldots r_i}{r_j}} \mid j \in [1, i] \}$, $\alpha^{r_1 \ldots r_i}$

Round n**:**

1. M_n selects $r_n \in Z_q^*$
2. $M_n \rightarrow$ All M_i: $\{ \alpha^{\frac{r_1 \ldots r_n}{r_i}.K_{in}} \mid i \in [1, n[\}$

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

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

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 $M_a \notin M$ 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 $M = \{M_1, \ldots, M_n\}$ 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 $\hat{r}_n \in Z_q^*$

2. $M_n \rightarrow M_{n+1}$: $\{ \alpha^{\hat{r}_n \frac{r_1 \ldots r_n}{r_i} K_{in}} \mid i \in [1, n] \}$, $\alpha^{\hat{r}_n r_1 \ldots r_n}$

Round 2**:**

1. M_{n+1} selects $r_{n+1} \in Z_q^*$

2. $M_{n+1} \rightarrow$ All M_i: $\{ \alpha^{\hat{r}_n \frac{r_1 \ldots r_{n+1}}{r_i} . K_{in} . K_{in+1}} \mid i \in [1, n+1] \}$

Upon receipt of the above, every M_i (M_{n+1} included) computes the new key as:

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

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 CLIQUE PROTOCOLS

A number of methods were developed during the last few years for the 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 ([MCJ97], [Low98], [DFG99], …). However, several tools allow to obtain proofs for unbounded systems at the cost of the interactive proof of several lemmas [Mea96] or of the risk of receiving no answer for some protocols [Son99]. Other approaches are based on the use of logics ([SvO94], [Pau98], …). They allow to obtain proofs for arbitrary size systems, but they often require error-prone formalization steps and does not provide the same support in pinpointing problems as the direct generation of counterexamples. Recently, “manual” approaches were presented, allowing to obtain fine-grained proofs for systems of any dimension, and even to analyze the interactions between protocols that can be executed concurrently (see [THG99a] 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 an honest session of the protocol is basically unbounded, what will intuitively result in dramatic state-space explosion problems. As we know, the only successful analysis of group protocols have been performed by theorem-proving approaches ([Pau97], [BS97]) 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 analyser by extending the power and scope of its theorem-proving capabilities [Mea00].

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 try to adapt ideas presented in the context of the strand space approach ([THG99b], …) 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 that is a subgroup of Z_p^* (p and q being large prime integers). 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 α (mod 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 at 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. $P = \left\{ \prod r_i^{e_i} \prod K_{jl}^{e_{jl}} \middle| e_i, e_{jl} \in Z \right\}$). The only sent elements are the ones of this type.

The behaviour 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 confidential.

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's session does not implies for any M_i the presence on the network 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 is hence 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 a honest M_i applied to the first element of the pair, and 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 from E to R, $(R,.)$ is the commutative group of which the elements of $Im(f)$ (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: The pair $(\alpha^x, \alpha^{xr_1K_{1n}^{-1}})$ will be represented by the element $r_1K_{1n}^{-1} \in R$.

The use of such a construction implies several hypothesis concerning the elements of G. We have actually to assume that any element of G can only

be computed in one way (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). In particular, we assume that $\alpha^{x+y} \neq \alpha^{z}$ (with $x, y, z \in P$) and, more generally, that a secret cannot be computed by combining elements of G (but only elements of E with elements of G). These hypotheses seem quite plausible given that we work within a large group 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 to capture the relation 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 the protocol. 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 proving correctness than in finding "optimal" attack sketches.

We are currently working on the development of a more rigorous framework to express these hypothesis and determine the measure in which they are idealization of the real capabilities of the intruder. We will now be looking at the ways that the intruder can use to manipulate our two sets of elements.

3.2 Intruder Capabilities

In [STW96], M. Steiner & al. showed that the problem of computing $\alpha^{x_1 x_2 \ldots x_n}$ from the view of the set of all $\alpha^{i_1 i_2 \ldots i_m}$ where $\{i_1, i_2, \ldots, i_m\}$ is a proper subset of $\{x_1, x_2, \ldots, x_n\}$ was equivalent to the DDH problem. This can be used to convince us that the combination of several elements of G sent during a session of the protocol cannot be of any use in order to compute a secret element (but we are not providing any proof of it at this time).

The only computation that can be useful for the intruder will then 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:

(1) If $e \in E_I$ and $r \in R_I$ then $r.e \in R_I$ 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: $G \rightarrow G$, $\alpha^x \rightarrow \alpha^{p.x}$ ($x, p \in P$), 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: $s(\alpha^x)=\alpha^{p.x}$, 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:

(2) If $s \in S : s(\alpha^x)=\alpha^{p.x}$, and $r \in R_I$ then $r.p \in R_I$ 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 Secrecy 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:

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

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: $s_1(\alpha^x) = \alpha^{p_1 \cdot x}$ and s_2: $s_2(\alpha^x) = \alpha^{p_2 \cdot x}$ 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: $s_1(\alpha^x) = \alpha^{p_1 \cdot x}$ and s_2: $s_2(\alpha^x) = \alpha^{p_2 \cdot x}$ 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 general 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. Suppression 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*: $s(\alpha^x) \rightarrow \alpha^{xe^i y}$ ($i \in Z$) 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 possible to find a solution of the system that meets all these constraints, then an attack on the protocol can be derived.

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 SUITE

In the first paragraphs, we will concentrate our study on the properties of the A-GDH.2 key generation protocol. Then, we will extend it by considering the concurrent use of the A-GDH.2-MA protocol.

4.1 Analysis of the A-GDH.2 Key Generation Protocol

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

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: $r_n K_{1n}$, …, $r_n K_{n-1n}$. The secrets are the following: $r_i K_{in}^{-1}$ for M_i ($1 \leq i < n$) and r_n for M_n.

Having so described the protocol, we will start our analysis by studying the implicit key authentication property, then we will turn to the perfect forward secrecy property, and finally to the resistance to known keys attacks property.

4.1.1 Implicit Key Authentication

In the study of this property, we can assume that the E_I -set is empty. If we follow the analysis scheme proposed above, we have now to express the linear system describing the "balance" of the variables of *E*. We will first look at the secrecy of $r_1 K_{1n}^{-1}$. 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:

$r_1 = 1, r_2 = 0, \ldots, r_{n-1} = 0$ (balance on $r_1, \ldots, r_{n-1}$)
$r_n K_{1n} + r_n K_{2n} + \ldots + r_n K_{n-1n} = 0$ (balance on r_n)
$r_n K_{1n} = -1,\ r_n K_{2n} = 0, \ldots,\ r_n K_{n-1n} = 0$ (balance on $K_{1n}, \ldots, K_{n-1n}$)

It can be observed that the summing of the *n*-1 last equations provides an inconsistency with the *n*-th equation. 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's definition. If we write this system in the case of multiple sessions of the protocol (for which *I* 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.

4.1.2 Perfect Forward Secrecy

In the study of this property, we will assume that E_I contains all long term keys K_{in}. If we apply the transformation suggested as second step of our proof-scheme, we can rewrite the set of services $S = \{r_i \mid i \in [2,n]\}$, $R_I = \{r_1\}$, and R_S as $\{r_i \mid (i \in [1,n]\}$. For each secret r_i, the resulting linear system has a trivial solution: $r_i = 1$. This solutions meets 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 $r_1 K_{1n}^{-1}$ 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 $S_n(M_n) = \alpha^{r_1 \dots r_n}$ and broadcast $\alpha^{r_1 \dots r_n K_{1n}}$, $\alpha^{r_1 r_3 \dots r_n K_{2n}}$, …, $\alpha^{r_1 r_2 \dots r_{n-2} r_n K_{n-1n}}$. Hence M_1 will be computing a wrong key: $S_n(M_1) = \alpha^{r_1^2 \dots r_n}$ while all other members of the group will compute $S_n(M_i) = \alpha^{r_1 \dots r_n}$ ($1< i \leq n$). 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 (that 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. 1. It seems to us that this is a much more awkward scenario. 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.

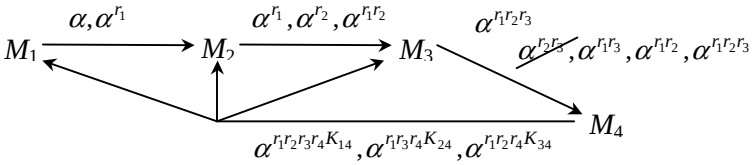

Figure 1. Attack against the Perfect Forward Secrecy Property

4.1.3 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 [AST00] 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:

$$S = \{ r_2,...,r_{n-1}, r_nK_{1n},...,r_nK_{n-1n}, r_2',...,r_{n-1}', r_n'K_{1n},...,r_n'K_{n-1n} \}$$
$$E_I = \varnothing$$
$$R_I = \{ r_1, r_1', r_1K_{1n}^{-1},...,r_{n-1}K_{n-1n}^{-1}, r_n \}$$
$$R_S = \{ r_1'K_{1n}^{-1},...,r_{n-1}'K_{n-1n}^{-1}, r_n' \}$$

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

$$r_iK_{in}^{-1} = 1,\ r_i = -1,\ r_i' = 1$$

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 α^{xr_i}.
2. The intruder replaces then the term $\alpha^{r_1...r_{i-1}r_{i+1}...r_nK_{in}}$ with α^x. Hence $S_n(M_i)$ will be equal to $\alpha^{xr_iK_{in}^{-1}}$. 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 inputs of the i-th round with $\alpha^{xr_iK_{in}^{-1}}$. M_i will therefore send $\alpha^{xr_iK_{in}^{-1}r_i'}$.
4. In the broadcast of the second run of the protocol, the intruder finally replaces the term intended to M_i with α^{xr_i} (obtained in the first step of our scenario). Hence $S_n'(M_i)$ will be computed as $\alpha^{xr_iK_{in}^{-1}r_i'}$ 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 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 [AST00]. 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 $1 \le i < n$, 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 $\alpha^{r_1 \ldots r_{i-1} r_{i+1} \ldots r_{n-1}}$ with $\alpha^{r_1 \ldots r_i \ldots r_{n-1}}$. Hence, all elements of the broadcast will be preserved except the one intended to M_i that will be equal to $\alpha^{r_1 \ldots r_n K_{in}}$. $S_n(M_n)$ will hence be equal to $\alpha^{r_1 \ldots r_n}$ 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 $\alpha^{r_1' \ldots r_{n-1}'}$ with $\alpha^{r_1 \ldots r_n K_{in}}$ and $\alpha^{r_1' \ldots r_{i-1}' r_{i+1}' \ldots r_{n-1}'}$ with $\alpha^{r_1 \ldots r_n}$. Hence M_n will broadcast $\alpha^{r_1 \ldots r_n K_{in} r_n'}$ and compute $S_n(M_n) = \alpha^{r_1 \ldots r_n K_{in} r_n'}$.

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 2. represents this scenario for i=1 and a group of four members.

Consequently, the resistance to known-key attacks seems problematic in this protocol.

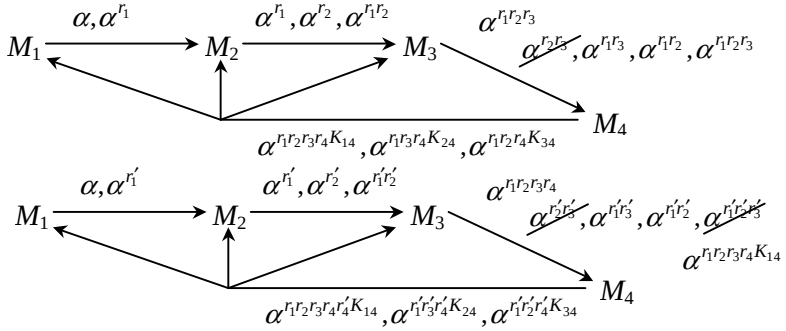

Figure 2. Attack against the Resistance to Known-Keys Property

4.2 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:

$S = \{ r_2,...,r_{n-1}, r_nK_{1n},...,r_nK_{n-1n},$ (A-GDH.2 Protocol)
$r_n\hat{r}_n, r_n\hat{r}_nK_{1n},...,r_n\hat{r}_nK_{n-1n}, \hat{r}_nK_{nn},$ (First round of A-GDH.2-MA)
$r_{n+1}K_{1n+1},...,r_{n+1}K_{n+1n+1} \}$ (Last round of A-GDH.2-MA)

$E_I = \varnothing$

$R_I = \{ r_1 \}$

$R_S = \{ r_1K_{1n}^{-1},...,r_{n-1}K_{n-1n}^{-1}, r_n,$ (A-GDH.2 Protocol)
$r_1K_{1n}^{-1}K_{1n+1}^{-1},...,r_{n+1}K_{n+1n}^{-1}K_{n+1n+1}^{-1} \}$ (A-GDH.2-MA Protocol)

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: $r_iK_{in}^{-1}$ $(1 \le i < n)$ can be obtained by combining the services (or ratios in the case of r_1) r_i, $r_n\hat{r}_n$ and $r_n\hat{r}_nK_{in}$ $(1 \le i < n)$. The other secrets can not 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 $\alpha^{r_1r_2...r_n\hat{r}_n}$ 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 a fourth member to the group is described in Fig. 3. Hence the Implicit Key Authentication property seems to become problematic when we consider the possibility of the use of the A-GDH.2-MA protocol in parallel with the A-

GDH.2 protocol. This security property being compromised, it does not seem useful to continue our analysis for the other properties.

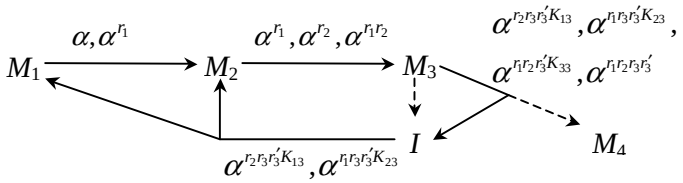

Figure 3. Attack against the Implicit Key Authentication Property

5. CONCLUSION

Throughout this paper, we presented the first steps of the development of a model for the analysis of the Cliques protocols. The reasoning in our model led us to pinpoint a number of unpublished flaws in the A-GDH.2 protocols suite, emphasizing the necessity to be able to reason systematically on security protocols, especially in contexts where active adversaries are to be considered. The scope of these flaws is summarized in the Table 1.

Table 1. Summary of the Flaws

Protocols Considered	Property Analysed	Number of Members Flawed
A-GDH.2	Implicit Key Authentication	No Attack Found
A-GDH.2	Perfect Forward Secrecy	*n*-1
A-GDH.2	Resistance to Known-Keys	1 (but parallel attacks possible)
A-GDH.2 and A-GDH.2-MA	Implicit Key Authentication	*n*-1

We are currently working on defining more precisely the attacks detectable (and those undetectable) with our model, on the incorporation of our "machinery" in more general models, 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.

References

[AST00] G. Ateniese, M. Steiner and G. Tsudik. *New Multi-party Authentication Services and Key Agreement Protocols*. IEEE Journal on Selected Areas in Communication, April 2000.

[BS97] J. Bryans, S. Schneider. CSP, PVS, and a Recursive Authentication Protocol. In DIMACS Workshop on Formal Verification of Security Protocols, 1997.

[DFG99] A. Durante, R. Focardi, R. Gorrieri. *CVS: A Tool for the Analysis of Cryptographic Protocols*. In Proceedings of the 12-th IEEE Computer Security Foundations Workshop, pp. 203-212, 1999.

[Low98] G. Lowe. *Casper: A Compiler for the Analysis of Security Protocols*. In Journal of Computer Security, Vol. 6, pp. 53-84, 1998.

[Mea96] C. Meadows. *The NRL Protocol Analyzer : an Overview*. In Journal of Logic Programming, Vol. 26(2), pp. 113-131, 1996.

[Mea00] 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 (WITS 2000), 2000.

[MCJ97] W. Marrero, E. Clarke, S. Jha. *A Model Checker for Authentication Protocols*. In Proceedings of the DIMACS workshop on design and formal verification of security protocols, 1997.

[Pau97] L C Paulson. *Mechanised Proofs for a Recursive Authentication Protocol.* In Proceedings of the 10th Computer Security Foundations Workshop, pp. 84-95. IEEE Computer Society Press, 1997.

[Pau98] L C Paulson. *The inductive approach to verifying cryptographic protocols*. In Journal of Computer Security, Vol. 6, pp. 85-128, 1998.

[Son99] 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.

[STW96] M. Steiner, G. Tsudik and M. Waidner. *Diffie-Hellman Key Distribution Extended to Group Communication.* In Proceedings of the 3rd ACM Conference on Computer and Communications Security, 1996.

[SvO94] P. Syverson, P. van Oorschot. *On Unifying Some Cryptographic Protocols Logics ».* In Proceedings of the IEEE Symposium on Research in Security and Privacy, pages 14-28, 1994.

[THG99a] F. J. Thayer, J. H. Herzog, J. Guttman. *Strand Spaces: Proving Security Protocols Correct.* In Journal of Computer Security, 7(2/3): 191-230, 1999.

[THG99b] F. J. Thayer, J. H. Herzog, J. Guttman. *Mixed Strand Spaces.* In Proceedings of the 12th Computer Security Foundations Workshop, pp. 83-89. IEEE Computer Society Press, 1999.