(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
5 October 2006 (05.10.2006)

PCT

(10) International Publication Number
WO 2006/103608 A2

(51) International Patent Classification: Not classified

(21) International Application Number:
PCT/IB2006/050909

(22) International Filing Date: 24 March 2006 (24.03.2006)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
05102580.7 1 April 2005 (01.04.2005) EP

(71) Applicant *(for all designated States except US)*: **KONINKLIJKE PHILIPS ELECTRONICS N.V.** [NL/NL]; Groenewoudseweg 1, NL-5621 BA Eindhoven (NL).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **TUYLS, Pim, T.** [BE/BE]; C/o Prof. Holstlaan 6, NL-5656 AA Eindhoven (NL). **KURSAWE, Klaus** [DE/BE]; C/o Prof. Holstlaan 6, NL-5656 AA Eindhoven (NL). **NEVEN, Gregory, K., R.** [BE/BE]; C/o Prof. Holstlaan 6, NL-5656 AA Eindhoven (NL).

(74) Agents: **GROENENDAAL, Antonius, W., M.** et al.; Prof. Holstlaan 6, NL-5656 AA Eindhoven (NL).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, LY, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SM, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

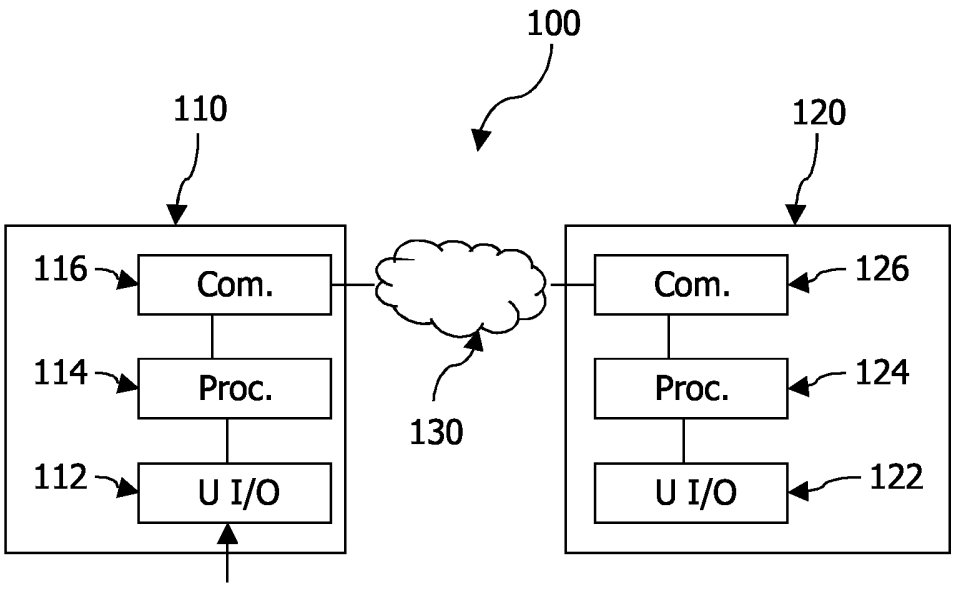
(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) **Title:** PRIVATE NEGOTIATION



(57) **Abstract:** A method of automatic agreeing on a set of attributes between a plurality of stations in a computer network. The method includes, in each station, determining at least one respective set of attributes that is acceptable for the station to be agreed with the other station. The attributes of the sets are selected from a predetermined list of attributes. The method uses a multi-party computation protocol to determine the agreed set based on the determined respective acceptable sets without revealing other information than a willingness to accept the agreed set.

WO 2006/103608 A2

Private negotiation

The invention relates to a method of automatic agreeing on a set of attributes between a plurality of stations in a computer network. The invention further relates to a computer program product for causing a processor to execute such method. The invention also relates to a system for agreeing on a set of attributes between a plurality of stations in a computer network. The invention further relates to a station for automatic agreeing on a set of attributes with at least one other station.

With the increasing amount of electronic data produced by day-to-day interactions, as well as the ability to link or otherwise process this data, the handling of privacy-sensitive data has emerged as an important field in computer security.

Consequently, substantial work has been done on defining privacy policies. The most common framework for privacy policies today is the Platform for Privacy Preferences Project (P3P), www.w3.org/p3p. P3P was designed with two goals in mind. Firstly, it creates transparency about an organization's privacy policy. It is required that both the overall policy and some details are provided in a human-readable format, so every user can find out about the policy in an easy way. Secondly, it allows automatic comparison of the policies. A user can thus define his personal privacy policy, and automatically get a warning if the policy of an organization he interacts with does not match (assuming the organization does provide a P3P policy). At its most basic level, P3P is a standardized set of multiple-choice questions, covering all the major aspects of a Web site's privacy policies. Taken together, they present a snapshot of how a site handles personal information about its users. P3P-enabled Web sites present their data-collection practices in a standardized, machine-readable, easy-to-locate manner. P3P-enabled browsers can "read" this snapshot automatically and compare it to the consumer's own set of privacy preferences. P3P enables Web users to understand what data will be collected by sites they visit, how that data will be used, and what data/uses they may "opt-out" of or "opt-in" to.

As an example, a P3P enabled browser may automatically fetch the P3P policy for a home page specified by the user. The policy might state that the only data the site

collects on its home page is the data found in standard HTTP access logs. The browser then checks this policy against the preferences of the user. For example, is this policy acceptable, or should the user be notified? If the user has selected a checkout page of an on-line shop then some additional information is required, such as the user's name, address, credit card number, and telephone number. For that page another P3P policy is available that describes the data that is collected here and states that this data will be used only for completing the current transaction. The browser examines this P3P policy. The user may have specified in the browser that the user wants to be warned whenever a site asks for the telephone number. In this case, the browser will pop up a message saying that this Web site is asking for the telephone number, and explaining the contents of the P3P statement. The user can then decide if this is acceptable and continue with the on-line order; otherwise the user can cancel the transaction. More complex criteria may also have been given to the browser. For example, the user may have indicated to be warned only if a site is asking for the telephone number and was going to give it to third parties and/or use it for uses other than completing the current transaction.

The current version of P3P does not deal with negotiation on a policy to be used. Rather, the server simply reveals its policy, and the client then proceeds with the interaction, or not. This works as long there is only one option that must be accepted or rejected by the user. In more complex situations where in principle one or more of the parties involved would be willing to interact with the other party under one of several possible options it is required to negotiate such an option, i.e. find a policy that is acceptable to all parties involved. The negotiated policy can be seen as a set of values for a given set of attributes (such as a users' telephone number).

It is an object of the invention to provide a method of automatically agreeing on a set of attributes between a plurality of parties in a computer network. It is a further object of the invention to provide a system for automatically agreeing. It is a further object of the invention to provide a station for automatically agreeing on a set of attributes with at least one other station.

To meet an object of the invention, the method of agreeing on a set of attributes between a plurality of stations in a computer network includes:

in each station determining at least one respective set of attributes that is acceptable for the station to be agreed with the other station by for each set of attributes

selected the attributes from a predetermined list of attributes and arranging the sets of attributes in a predetermined order;

using a predetermined multi-party computation protocol to evaluate in each station the respective ordered lists of acceptable sets without revealing to the other station any information other than a willingness to accept the agreed set, and to determine the agreed set based on the evaluation.

In the current version of P3P, the client's privacy is protected as the client does not reveal any data outside the ones automatically revealed in contacting the server. If negotiations are introduced, however, the problem of privacy is not that simple, especially if there is reason to protect the privacy of the server as well. Usually, it is assumed that both sides somehow agree on a common policy specifying what data will be transmitted and how sensitive data should be handled. In very privacy-sensitive settings, this negotiation is complicated by the additional effect that a person's privacy preferences may already give away information about that person. Since the vast majority of the population is still willing to reveal seemingly innocent data (e.g. their monthly consumption of alcohol), a person that considers this piece of data as sensitive might quickly raise suspicion and consequently be treated with a worst-case assumption (e.g. that he's an alcoholic). Particularly, in a peer-to-peer scenario, e.g. between two consumers or two companies, it is important to protect the privacy of both sides. The method according to the invention enables automatic agreement on a set of attributes by applying a secure multi-party computation protocol for negotiating the set of attributes. Using such a protocol has the advantage that an agreed set can be determined (if that exists) without revealing to the other party information other than a willingness to accept the agreed set. For example, a server can thus not detect the privacy preferences of a client station whereas still an agreed set can be negotiated. Thus, privacy settings are themselves considered as private data. It will be appreciated that the scope of the invention is much broader than negotiating a policy between a client station and a server. In principle, the invention covers automatic negotiation between at least two computing devices where each device has at least one acceptable set of attributes from a predetermined list of attributes. The sets that are not agreed are not revealed.

According to the measure of the dependent claim 2, the multi-party computation protocol is based on a homomorphic encryption system. Using such an encryption system makes the evaluation fast.

According to the measure of the dependent claim 3, the homomorphic encryption system is an El Gamal or Paillier type of encryption. Both are established suitable

encryption systems where El Gamal has the additional advantage of an efficient generation of the encryption keys being used.

According to the measure of the dependent claim 4, the method includes the step of representing for each stations the acceptable sets of attributes using respective truth tables including a preference column that for each possible set of attributes of the predetermined list of attributes includes information on a level of acceptability of the set. This is an effective way of defining the sets of attributes that can be understood by the users and enables automatic further processing of the information. The acceptability may simply be two-valued (acceptable, not-acceptable), but more valued options are possible too.

According to the measure of the dependent claim 5, the method includes selecting for a first one of the stations a generating subset of acceptable sets of attributes where an acceptable set of attributes of the first one of the stations is a generating subset if there is no other acceptable set of attributes that fully includes the acceptable set of attributes; and selecting for a second one of the station a generating subset of acceptable sets of attributes where an acceptable set of attributes of the second one of the stations is a generating subset if there is no other acceptable set of attributes that includes not at least one of the attributes of the acceptable set of attributes. Selecting the so-called generating subsets can significantly reduce the number of sets that need to be evaluated, increasing speed. The first station plays in the negotiation a 'client' role. If the client station is willing to accept set A (e.g. to reveal the attributes on set A to the other station) then automatically the client station is also willing to accept a related set B with less of those attributes $(B \subset A)$. In this scenario B is not a generating subset. A candidate acceptable set for the first station is thus a 'generating subset' if it is not itself a subset of a larger acceptable set, where larger means that that set includes all attributes of the candidate set and at least one more. The second station plays in the negotiation a 'server' role. If the server station is willing to accept set A (e.g. the client station reveals the attributes on set A to the other station) then automatically the server station is also willing to accept a related set B with more of those attributes $(B \supset A)$. In this scenario again B is not a generating subset. A candidate acceptable set for the second station is thus not a 'generating subset' if there is smaller acceptable set, where 'smaller' means that all the attributes of the smaller set are also member of the candidate set. If this is not the case, then the acceptable set is a generating subset.

According to the measure of the dependent claim 6, includes forming a Boolean circuit implementing a process of matching the generating subsets, where the

generating subsets the respective preference columns for the generating subsets are used as input and an index representing a mutually acceptable set of attributes is generated as output. A Boolean circuit is a representation that can easily be evaluated by a secure multi-party computation protocol.

According to the measure of the dependent claim 7, the predetermined multi-party computation protocol is used to evaluate the Boolean circuit. This is an effective way of achieving a secure multi-party agreement.

According to the measure of the dependent claim 8, the method includes using a predetermined selection criterion to select the agreed set among a plurality of mutually acceptable sets of attributes. In this way a choice can be made between multiple mutually acceptable candidate sets. Any suitable selection criterion may be used, such as the first acceptable set on the list, assigning weights to preferred sets, etc.

According to the measure of the dependent claim 9, the method further includes agreeing on a set of obligations between the stations; each of the stations being associated with at least one respective set of obligations selected from a predetermined list of obligations; the method including using the multi-party computation protocol to determine the agreed set of obligations without revealing other information than a willingness to accept the agreed set of obligations. In this way not only a set o attributes can be agreed but also the condition under which these attributes may be used. Again, conditions other than the agreed conditions for the agreed set are not revealed to the other party. This enables automatic negotiations of complex agreements.

To meet an object of the invention, a computer program product is provided for causing a processor to execute the method as claimed in claim 1.

To meet an object of the invention, a system for automatic agreeing on a set of attributes includes a plurality of stations arranged to communicate through a computer network, where each station includes:

respective means for determining at least one respective set of attributes that is acceptable for the station to be agreed with the other station by for each set of attributes selecting the attributes from a predetermined list of attributes and arranging the sets of attributes in a predetermined order; and

respective means for executing a predetermined multi-party computation protocol to evaluate in each station the respective ordered lists of acceptable sets without revealing to the other station any information other than a willingness to accept the agreed set, and to determine the agreed set based on the evaluation.

To meet an object of the invention, a station for automatic agreeing on a set of attributes with at least one other station and arranged to communicate through a computer network with the at least one other station, includes means for determining at least one set of attributes that is acceptable for the station to be agreed with the at least one other station by for each set of attributes selecting the attributes from a predetermined list of attributes and arranging the sets of attributes in a predetermined order; and means for participating in a predetermined multi-party computation protocol to evaluate the ordered list of acceptable sets without revealing to the at least one other station any information other than a willingness to accept the agreed set, and to determine the agreed set based on the evaluation.

These and other aspects of the invention are apparent from and will be elucidated with reference to the embodiments described hereinafter.

In the drawings:

Fig. 1 shows a block diagram of a system in which the invention may be employed;

Fig. 2 shows a flow diagram of the method according to the invention;

Fig. 3 shows a Boolean circuit representing the acceptable sets to be securely evaluated; and

Fig. 4 illustrates an extension to the Boolean circuit incorporating conditions.

The method and system according to the invention deal with multiple parties in a computer network agreeing on a set of attributes to be used. The invention will be described for two parties. Persons skilled in the art can easily extent the principle to more than two parties using the multi-party computational protocol for more than two parties. The negotiation (agreement) is performed automatically. As such the parties can be seen as 'computing devices', or 'stations'. Since such a device is usually controlled by a human, it is assumed that the negotiating is performed on behalf of the human. In the description, the term 'party' will thus sometimes be used as referring to a user on which behalf the station operates.

Fig. 1 shows a block diagram of an exemplary system in which the invention may be used. The system 100 includes at least a first station 110 and a second station 120. The stations (parties) can communicate via a computer network 130, such as Internet. For the

communication the stations are equipped with suitable hardware and/or software, shown as block 116 and 126, respectively. Such hardware/software is well-known and will not be described here any further. The stations are equipped with a respective user interface 112, 122 for interaction with a user or human operator. Any suitable user interface may be used, such as display and keyboard/mouse. In the embodiment of Fig. 1, the core of the processing defined by the invention is performed by a digital processor, such as a general purpose micro-processor or signal processor. Preferably, at least part of the secure multi-party computation is performed in a secure module which in itself is well-known. Such a module may but need not include a specialized cryptographic processor which is optimized for fast execution of cryptographic operations. The stations itself may be based on conventional computer platforms such as a desktop PC, a server computer, or even mobile computing platforms, such as PDAs or advanced mobile phones. It will be appreciated that the stations may further be equipped with conventional means such as a main memory (RAM-type), permanent memory (e.g. ROM, hard disk, flash, etc.), internal buses, etc. The memory may be used to store, for example, the attribute sets.

Fig. 2 shows an exemplary structure of steps that can be performed by the processors 114, 124. Each of those steps may be implemented in a separate software procedure, but may also be combined in less software modules. Each of the stations in principle performs the same operations. In step 210, of all possible sets of attributes the subset of attributes is determined that is acceptable to the station. Typically, a human user/operator indicates this to the station via the user interface 112, 122, respectively. As will be described in more detail below, in a preferred embodiment shown in step 220 the number of acceptable sets is reduced to those sets that can be regarded as the generating subsets. This reduces the number of sets that need to be evaluated. Step 230 shows a further optional step wherein additional conditions are determined for each of the stations as will be described in more detail below. In step 250 the secure multi-party computation protocol is executed to evaluate the agreeable attribute sets and optional conditions in order to find a set on which the stations agree. For this evaluation to be performed effectively using a known secure multi-party computation protocol, first in step 240 a Boolean circuit is built representing the attribute sets (and optional conditions) to be evaluated. The Boolean circuit is a 'virtual' circuit in the sense that the gates of the circuit need not be implemented using actual electronic gates. In fact, known secure multi-party computation protocols may be used to evaluate the logical circuit in a known way.

Notations

Let $N=\{0,1,2,...\}$ be the set of natural numbers. Let $Z=\{...,-2,-1,0,1,2,...\}$ be the set of integers. If $k\in N$, then 0^k is the k-bit string of k zeros, and 1^k is the k-bit string of k ones. The individual bits of a k-bit string x are indicated as $x_0,\ldots,x_{k-1}$. If $b\in\{0,1\}$ then $\neg b$ is the negation of b; if $b_1,b_2\in\{0,1\}$ then $b_1\wedge b_2$ is the logical conjunction (AND) of b_1 and b_2, and $b_1\vee b_2$ is the logical disjunction (OR) of b_1 and b_2.

Definitions

In the remainder, one of the parties that is offering a service will be referred to as 'server' whereas the other party requesting the service will be referred to as 'client' or 'user'. The 'client' and 'server' may be formed by a client station and server station, respectively, as for example known from Internet. Within the scope of the invention also lie other types of network or interaction schemes. For example, the two parties may be involved in a peer-to-peer (P2P) interaction through the Internet. Also in such a scenario one of the stations requests a 'service' from the other station. It should be noticed that in the system/method according to the invention an agreement is made between at least two stations. The details of the agreement and the purpose of the agreement are in itself not relevant for the invention. For example, the agreement could result in a transfer of data (e.g. audio/video) from a server station to a client station, it could result in the sale of a product or delivery of a service by the owner of the server to the user of the client, etc.. In the remainder for the purpose of explanation typically it will be described that the server itself provides a service to the client. As will be clear, that in itself is not essential for the invention and is purely for explanation. In general, the agreement could be seen as part of a contract between two parties which, according to the invention, is established using electronic means and in a secure way so that no third party can see the negotiation and not even the parties involved can see what the other party would be willing to accept other than the final outcome of the process. So, the parties can keep all their cards on the table and only need to reveal the card that they known the other party will accept.

By preferences of a user (= user of the client station) is meant a strategy defining which attributes (e.g. credit card number, address, ...) the user is willing to reveal in turn for getting a certain service. A server's preferences defines the set of attributes required from a user in turn for offering a certain service. The negotiation according to the invention refers to the process of finding out whether the attributes that the user wants to reveal correspond to the set of attributes that the server requires. A combination of attributes is a

matching subset if it is acceptable to both the client and the server. More formally, the problem of subset matching is defined as follows: Let Â be a predetermined set of attributes, and let S be a totally ordered set of scores with least element 0. Preferences over the set of attributes Â are described by a function $f:2^{\hat{A}} \to S$ that assigns to each subset of attributes $A \subseteq \hat{A}$ a score $s \in S$, indicating the client's willingness to reveal the combination of attributes $\hat{A}$ (in the case of client preferences), or indicating the server's inclination to accept that combination of attributes as sufficient to access the service (in the case of server preferences). A matching function $M:2^{\hat{A}} \times S \times S \to S$ assigns to $A \subseteq \hat{A}$ a matching score based on $\hat{A}$, the client's willingness $f(A)$ and the server's acceptance $g(A)$. A combination of attributes $A \subseteq \hat{A}$ is said to be a matching subset with respect to client preferences f, server preferences g and matching function M if $M(A, f(A), g(A)) > 0$. A best matching subset is a subset $A \subseteq \hat{A}$ for which $M(A, f(A), g(A))$ is maximal.

For the sake of clarity, in the remainder the embodiment is described in more detail where $S = \{0,1\}$, meaning that a client is either willing to reveal a combination of attributes (e.g. indicated using a binary 1) or not (e.g. indicated using a binary 0). Analogously the same scoring is used for the server indicating whether a server accepts a combination of attributes or not. It will be appreciated that also more than a two-valued score is possible. For example, a client is either willing to always reveal a combination of attributes (e.g. indicated using a binary 11), never to reveal (e.g. indicated using a binary 00), highly reluctant (binary 01), slightly reluctant (binary 10). As will be clear then a larger score set is used. If not all options of a multi-bit score would be used, dummy values may be used.

By private subset negotiation is meant a matching process between the client and the server where during the process they learn nothing about each other's preferences (i.e. agreeable subsets) except whether a matching subset exists, and possibly what that matching subset is.

Using Truth Tables

Let $\hat{A} = \{a_0, \ldots, a_{n-1}\}$ be the set of the client's attributes (e.g. a_0 = credit card no., a_1 = bank account no., a_2 = telephone no, a_3 = address, a_4 = birthday,…). Each $x \in \{0,1\}^n$ is associated with a subset of attributes $A(x) = \{a_1 : x_1 = 1, 0 \le i < n\} \subseteq \hat{A}$. In the following, the special case with score set $S = \{0,1\}$ is explained further where the matching function M gives 1 if and only if $f(A) = g(A) = 1$. The client's preferences can then

equivalently be modeled as a boolean function $f:\{0,1\}^n \rightarrow \{0,1\}$, where $f(x)=1$ if the client is willing to agree (and reveal) the combination of attributes $A(x)$, and is 0 if he'd rather not agree this combination. Likewise, the server's preferences can be modeled as a boolean function $g:\{0,1\}^n \rightarrow \{0,1\}$ such that $g(x)=1$ if agreeing attributes $A(x)$ is sufficient to access the service, and is 0 if it isn't. A matching subset is an assignment $x \in \{0,1\}$ such that $f(x)=g(x)=1$.

The functions $f(x)$ and $g(x)$ are preferably represented through their truth tables, as illustrated in the following tables.

x_0	x_1	...	x_{n-1}	$f(x)$
0	...	0	0	0
0	...	0	1	1
0	...	1	0	0
0	...	1	1	0
...	...	...	...	...
1	...	1	1	0

x_0	x_1	...	x_{n-1}	$g(x)$
0	...	0	0	0
0	...	0	1	1
0	...	1	0	1
0	...	1	1	1
...	...	...	...	...
1	...	1	1	1

It will be appreciated that, in the shown form, the top row shows the possible attributes. Each other row represents an attribute subset. An attribute a_i is part of an attribute subset only if $x_i=1$. The last column indicates for each respective row (i.e. each for each possible attribute subset) whether the client (Fig. 2A) or the server (Fig. 2B) is willing to accept that subset. This is indicated by the preference functions $f(x), g(x)$, respectively. Deciding whether a matching subset exists comes down to finding a row (attribute subset) with a 1 in the rightmost column of both truth tables. It will be appreciated that a match can only be determined if all stations assume the same ordered list of possible attributes and attribute subsets to be agreed.

In the example above, the second subset in the tables $x=0,\ldots,0,1$ would be a match, which corresponds to showing only attribute a_{n-1}.

In an embodiment according to the invention, the matching process is performed using a Boolean circuit with as input the client's input in the form of the bit string $f \in \{0,1\}^n$ (i.e. the rightmost column of the truth table of $f(x)$), and the server's input in the

form of the bit string $g \in \{0,1\}^n$ (i.e. being the rightmost column of the truth table of $g(x)$). The output of the circuit is an index $i \in Z_{2^n}$ such that $f_i = g_i = 1$. However, the size of this circuit (in number of inputs and number of gates) is $O(2^n)$, making this approach less suitable for evaluation through secure two-party protocols since this would quickly get too slow. To this end, in an embodiment of the invention the number of attribute subsets that need to be evaluated is reduced.

Generating Subsets

A more compact description of the client's and server's agreeable subsets is obtained by observing that in most real-world cases if the client is willing to show the combination of attributes A, then it is natural to assume that the client is willing to show any subset of these attributes as well. This is reflected by the preference function f being a monotonically decreasing boolean function. This means that if $f(A) = 1$ and $B \subseteq A$, then also $f(B) = 1$. Likewise, if showing attributes A is sufficient to access a service at the server, then so should be any combination $B \supseteq A$. This is being reflected by that usually g is monotonically increasing, meaning that if $g(A) = 1$ and $B \supseteq A$, then $g(B) = 1$. In an embodiment according to the invention, this is used by determining the so-called generating subsets for both the client and the server respectively. Mathematically those generating subsets are expressed as follows. Let $F_1,...F_a \in 2^{\hat{A}}$. The set $\mathrm{F} = \{F_1,\ldots,F_a\}$ is a *set of generating subsets* for a monotonically decreasing boolean function f iff (if and only if) for all $A \subseteq \hat{A}$:

$$f(A) = 1 \Leftrightarrow \exists i \in \{1,\ldots,a\} : A_i \subseteq F$$

Analogously, the set $G = \{G_1,...,G_b\} \subset 2^A$ is a *set of generating subsets* for a monotonically increasing function g iff for all $A \subseteq \hat{A}$

$$g(A) = 1 \Leftrightarrow \exists j \in \{1,\ldots,b\} : G_j \subseteq A$$

These generating subsets are a very compact representation of the client's and server's preferences. Preferably, they are generated automatically by analyzing the preferences given by a user (i.e. the agreeable subsets). Alternatively, via a user interface the user of the clients station or server, respectively, may be requested to specify this set of generating subsets. A human user normally has a good idea of the "minimal" combinations of attributes he doesn't want to reveal, independent of whether additionally even more attributes being revealed. For example, he may be reluctant to simultaneously show his credit

card number and his mother's maiden name (the latter is sometimes used as a backup secret to reactivate lost cards), and having to add his favorite color will not make him change his mind. Analogously, the server knows the minimal information that he needs from users (e.g. name and either email address or phone number), but he won't mind extra attributes being given. The same principle holds for any form of contract negotiations.

Using the notation of generating subsets, finding a match is equivalent to finding a set of attributes $A \subseteq \hat{A}$ such that

- $\forall F_1 \in F : F_1 \, is \, not \subset \hat{A}$, and
- $\exists G_j \in G : G_j \subseteq \hat{A}$.

Without loss of generality, it can be assumed that the matching subset is one of the server's generating subsets, if a match exists. This is the match with the smallest number of shown attributes). Therefore, the condition for the matching subset A can be written more compactly as the set of attributes $A \in G$. such that $\forall F_1 \in F : F_1 \, is \, not \subset \hat{A}$.

Truth tables and generating subsets are just different ways of describing the same preferences (acceptable subsets). The relationship between both can be expressed as follows: $f(x) = 0$ if and only if there exists a subset $F_i \in \mathrm{F}$ such that $F_i \subseteq A(x)$, and that $g(x) = 1$ if and only if there exists a subset $G_i \in \mathrm{G}$ such that $G_i \subseteq A(x)$. Casting this into a boolean formula gives

$$\neg f(x) = \bigvee_{F_i \in \mathrm{F}} \left(\bigwedge_{a_{j \in F_i}} x_j \right),$$

$$g(x) = \bigvee_{G_i \in \mathrm{G}} \left(\bigwedge_{a_{j \in G_i}} x_j \right)$$

showing that the generating subsets are actually the terms of $\neg f(x)$ and $g(x)$ when written in disjunctive normal form (DNF).

The Boolean Circuit

Here will be shown that the matching process for the generating subsets approach given in Fig. 3 can be described as a Boolean circuit. Inputs to the circuit are the strings describing the generating subsets. Later it will be explained how this circuit can be evaluated in a private way. During the evaluation in each station, the station has knowledge of its own strings (in plaintext form) but has only access to the strings of the other station in encrypted form. It will be appreciated that a same circuit can be constructed for a full set of possible attribute sets, but that that in most practical cases would give a too large circuit for

fast evaluation. The client's input consists of the generating subsets $F_1,\ldots,F_a$ encoded as n-bit strings, where the j-th bit of F_i is 1 iff attribute $a_j \in F_i$. The server's input consists of the subsets $G_1,\ldots,G_b$ in the same encoding. It is noted that the values of a and b may leak information about the complexity of the respective client's and server's preferences. To avoid such leaking it is preferred that the circuit is designed for some predetermined maximum values of a and b. The client and server assign then arbitrary values to unused F_i and G_j entries, but distinguish "real" subsets from "dummy" subsets, for example by setting the additional input bits f_i and g_j to 1 or 0, respectively. The output of the circuit is the encoding of a matching subset M, and a bit e indicating whether a matching subset exists ($e=1$) or not ($e=0$). In the exemplary circuit, when no matching subset is found, M takes the value $0,\ldots,0$.

Gates with multiple fan-in in Fig. 3 can be implemented as a cascade of binary gates, or as a balanced tree of binary gates. Both options are equivalent in the total number of gates, but the latter option gives a better efficiency in terms of communication rounds. The thick lines in Fig. 3 represent buses, which are essentially collections of parallel wires to carry words, rather than individual bits. Thick gates represent bitwise operations on words, e.g. the bitwise AND of n-bit words $x,y \in \{0,1\}^n$ is the n-bit word $z=(x_0 \wedge y_0,\ldots,x_{n-1} \wedge y_{n-1})$.

A first layer L1 of the circuit checks, for all $j=1\ldots b$, whether G_j is a suitable candidate for a matching set, meaning that G_j does not conflict with any of the client's F_i sets. The wire labeled c_j in Fig. 3 carries a one if G_j is a candidate set, and a zero if not.

The circuit first computes, for all $i=1,\ldots,a$ and $j=1,\ldots,b$, whether the client's set F_i is "compatible" with the server's set G_j, meaning that F_i is not $\subseteq G_j$. This is equivalent to checking that there exists at least one attribute $a \in \hat{A}$ such that $a \in F_i$ but $a \notin G_j$. The output word of the bitwise AND gates in Layer 1 contains a one-bit for each such attribute. OR-ing the n bits of this word together gives 1 iff at least one such attribute exists. By including $\neg f_i$ in the OR, the output of the OR-gate is forced to 1 for dummy input sets F_i, representing the fact that subset G_j should never be rejected because of a conflict with a dummy set. The rightmost AND gates in Layer 1 decide, for all $j=1,\ldots,b$, whether

G_j is a candidate matching subset, meaning that it was "compatible" with all (non-dummy) client reluctancy sets. By including g_j in the AND gate, c_j is forced to 0 if $g_j = 0$, representing the fact that a dummy server set G_j can never be a suitable candidate.

In a preferred embodiment, a second layer L2 of the circuit finds the candidate subset with the lowest index and outputs it as the matching set. To select the match with the smallest index, the circuit uses intermediate variables e_j that are 1 iff a matching set exists among $G_1,\ldots,G_j$. The next value for e_j is computed as $\neg e_{j-1} \wedge c_j$, and the output bit $e = e_b$ is one iff a matching set was found. AND-ing c_j with $\neg e_{j-1}$ ensures that the only non-zero bit coming out of any of the leftmost AND gates in Layer 2 is on the wire corresponding to the first match. The final gates of the circuit encode a matching set onto the output bus M. Assume a match was found, and let j' be the index of the first match. Then the output of the j'-th bitwise AND gate in Layer 2 is $G_{j'} \wedge 1^n = G_{j'}$, while for all other AND-gates it is $G_{j'} \wedge 0^n = 0^n$. The final bitwise OR-gate sets the output M to $G_{j'} \vee 0^n \vee \ldots \vee 0^n = G_{j'}$. If no matching set was found, then M is set to 0^n, leaking no information about the server's preferences except the fact that no match exists with the given client preferences.

It will be appreciated that layer 2 can be replaced by another selection mechanism than selecting the commonly acceptable subset with the lowest index. For example, the stations may assign weights to subsets acceptable to the stations. Layer 2 would then need to be redesigned to select the candidate with maximum weight. Designing this lies well within the skills of a person skilled in the art.

Subset Matching with Obligations

In a preferred embodiment, the method and system according to the invention are extended to allow the client to express demands concerning certain attributes, such as to delete the data after a certain time, not to forward it to third parties, ... The server expresses which promises he's willing to make for which attributes. A matching subset is then defined as a combination of attributes such that (1) they suffice to access the service, (2) the client is willing to reveal them, and (3) the server is willing to comply with the client's demands related to the revealed attributes. This extension is described for the preferred embodiment with ordered sets but can equally well be applied to the unordered set scenario. Let O be a set of obligations. The client's demand function $d:\hat{A} \rightarrow 2^{\circ}$ associates to each attribute a set of obligations that the client demands from the server when revealing that attribute. The server's

willingness function $w:\hat{A}\rightarrow 2^{\circ}$ maps an attribute to the set of obligations that the server is willing to respect for that attribute. A combination of attributes $A\subseteq\hat{A}$ is now said to be a matching subset with respect to client preferences f, server preferences g and matching function M, demand function d and willingness function w if $M(A,f(A),g(A))>0$ and $\forall a\in A: d(A)\subseteq w(A)$. A best matching subset is a subset $A\subseteq\hat{A}$ for which $M(A,f(A),g(A))$ is maximal.

As above, the description only focuses on the special case where $S=\{0,1\}$, where f and g are monotonically decreasing, respectively increasing, boolean functions, and where the result of the matching function $M(A,f(A),g(A))=1$ iff $f(A)=g(A)=1$. Let $O=\{o_0,...,o_{m-1}\}$ be the set of promises that the client can demand for each attribute (e.g. o_0 = "delete after session", o_1 = "delete after one year", o_2 = "do not forward to third parties").

Fig. 4 illustrates the modification that this extension gives to the circuit of Fig. 3. The client's demand function and the server's willingness function are described by additional inputs $D_i=d(a_i)$ and $W_i=w(a_i)$ for $i=0\ldots m-1$, respectively, encoded as m-bit strings. Apart from a bit e indicating whether a match exists and the matching subset M, the circuit now also outputs the obligations $O_0,\ldots,O_{n-1}\subseteq O$ that the server has to adhere to for attributes $a_0,\ldots,a_{n-1}$. The circuit in Fig. 4A is inserted between Layers 1 and 2 in Fig. 3 on each wire $c_j, j=1\ldots b$, replacing c_j with a bit $c_{j'}$ before passing it to Layer 2. For each candidate set G_j, this subcircuit computes a bit $c_{j'}$ indicating whether the server is also willing to make all promises that the client requires for attributes in G_j. The first bitwise AND gate simply encodes G_j on the bus if G_j is a candidate ($c_j=1$), or encodes a string of zeroes if not ($c_j=0$). This encoding is split up in n separate wires representing whether attribute $a_i\in G_j$ or not, for $i=0\ldots n-1$. For each attribute a_i, the second column of bitwise AND gates outputs an m-bit word with a 1 on each wire representing an obligation $o\in O$ that the client demands ($o\in D_i$) but that the server is not willing to make ($o\notin W_i$). If any such promise exists in any of the attributes in G_j, then subset G_j is not a match. This is exactly what is computed by the OR gate. The final NOT gate inverts this result so that $c'_j=1$ if and only if G_j is a candidate match for which additionally the promises work out.

The second circuit in Fig. 4B is to be attached to the right of the circuit in Fig. 3. If attribute $a_i \in M$, then the bitwise AND gate encodes the client's demands D_i onto the output promises O_i, or it encodes the all-zeroes string if $a_i \notin M$.

Secure Evaluation of a Boolean circuit

Above, the private subset negotiation problem has been turned into a problem of the secure evaluation of a function that can be described as a circuit. The circuits that have to be evaluated are built up from basic gates in casu NOT, OR and AND gates. Below protocols are presented for the secure evaluation of those gates within the model of secure two-party computation; i.e. two parties are considered who evaluate these gates without revealing anything about their input (except the information that leaks from the output of the function). A distinction is made between protocols having only encrypted inputs and those having an encrypted and an unencrypted input. An unencrypted input refers to a private input of a user. The execution of the protocol should not reveal any information on this input. Encrypted inputs are inputs containing a message unknown to both players. Often they are the result from a previous (intermediate) computation whose result should remain unknown to both users. Clearly, the execution of the protocol should not reveal anything more on the message hidden by the encryption than what is revealed by the result of the protocol. In the remainder first background on preferred encryption protocols is given, followed by a description how the gates of the circuit can be securely evaluated. The remainder will focus on using a homomorphic crypto-systems for the evaluation. However other suitable techniques, such as garbled circuit approach or oblivious transfer, may be used and fall within the knowledge of a person skilled in multi-part computation protocols.

Homomorphic Encryption

Given an encryption function E, a public key p with corresponding secret key s, and a message m, we denote by $E_p(m)$ the encryption of m with the public key p. An encryption function E is called additively homomorphic if for all messages m_1 and m_2, we have $E_p(m_1 + m_2) = E_p(m_1)E_p(m_2)$. Well-known examples of homomorphic cryptosystems are the El Gamal cryptosystem and the Paillier cryptosystem. In this description, the El Gamal cryptosystem is used, but the Paillier system may also be used. The El Gamal system has the advantage that key generation can be done very efficiently. It is therefore very well-suited for the peer-2-peer (P2P) setting. A brief description of the homomorphic version of the El Gamal cryptosystem follows. Let $G =< g >$ denote a finite

cyclic (multiplicative) group of prime order q for which the Decision Diffie-Hellman (DDH) problem is assumed to be infeasible: given $g^x, g^y, g^z \in_R G$, it is infeasible to decide whether $xy \equiv z \pmod{q}$. This implies that the Computation Diffie-Hellman (CDH) problem (i.e. compute g^{xy} given $g^x, g^y \in_R G$) is infeasible as well. In turn, this implies that the Discrete Log (DL) problem, which is to compute $\log_g h = x$ given $g^x \in_R G$, is infeasible.

The public key of the El Gamal cryptosystem is an element $h \in G$ and the encryption of a message $m \in Z_q$ is given by the pair $(a,b) = (g^r, h^r g^m)$ where $r \in_r Z_q$. The secret key s is given by $s = \log_g h$. Given the private key s, decryption of the ciphertext $(a,b) = (g^r, g^m h^r)$ is performed by first calculating $b/a^s = g^m$, and then solving for $m \in Z_q$. In order to solve this DL-problem (which we assumed to be infeasible) we consider only messages m belonging to a sufficiently small domain $m \subseteq Z_q$. In this description, we take $M = \{0,1\}$.

The multiplication of two cipher texts (for the same public key) (a,b) and (a',b') is defined by $(a,b)(a',b') = (aa',bb')$. It readily follows that this encryption scheme is additively homomorphic for this multiplication. Note that a message m has many encryptions under the same public key because of the randomness r involved. By multiplying $E_p(m)$ with $E_p(0)$ one obtains a new encryption of m under the same public key. Under the DDH assumption, this cryptosystem is semantically secure.

For ease of notation, $[m]$ is used to denote an encryption of the message m under some understood public key. In this notation we have $[x][y] = [x+y]$ and $[x]^y = [xy]$.

Threshold decryption

In an (n,t) threshold cryptosystem, the private key is distributed over n parties such that only coalitions of size at least t parties can decrypt the message hidden inside a ciphertext. In this description here a $(2,2)$ threshold cryptosystem is used where encryptions are computed with a common public key h but decryption is performed by running a protocol between the two involved parties. Every party holds a share $s_1 \in Z_q$ of the private key $s = s_1 + s_2 = \log_g h$, where the corresponding value $h_i = g^{s_i}$ is public. As long as both parties take part, decryption will succeed, whereas only one party is not able to decrypt successfully. Assuming that the parties have already obtained their shares of the secret key

(through a distributed key generation protocol, explained below), decryption is performed as follows. For decryption of the ciphertext (a,b), the players P_1 and P_2 produce a decryption share $d_i = a^{s_i} (i = 1,2)$ along with a proof that $\log_a d_i = \log_g h_i$. Assuming without loss of generality that both players produce correct decryption shares, the message is recovered from solving $g^m = b/a^s$ (where a^s is obtained from $d_1 d_2$) for m. Finally, it is checked whether $m \in \{0,1\}$. If this does not hold decryption fails.

In case both parties need to obtain the decrypted value, the protocol has to be run in a fair way. A protocol for this is given in "Berry Schoenmakers and Pim Tuyls, "Practical two-party computation based on the conditional gate", Advances in Cryptology – ASIACRYPT 2004, volume 3329 of Lecture Notes in Computer Science, pages 119-136, Springer-Verlag, Berlin, Germany, 2004", hereby incorporated by reference. This article will be referred to as [ST04]. In case only one of the parties sends his share to the other party, a threshold decryption protocol with private output is obtained.

Distributed Key Generation

In order to set up the key-generation in a P2P situation, the users have to run a Distributed Key Generation (DKG) protocol. Any suitable protocol may be used. In a practical protocol, in the first step both parties broadcast a Pedersen commitment $c_i = g^{s_i} h^{r_i}$, with $s_1, r_1 \in_R Z_q$ along with a proof of knowledge for s_i, r_i. In the second step, both parties broadcast r_i along with a proof of knowledge of $\log_g h_i$, where $h_i = c_i / h^{r_i}$. The joint public key is $h = h_1 h_2$, with corresponding private key $s = s_1 + s_2$. In many practical cases, a more light-weight one-round protocol may also be used. Then, both players broadcast $h_i = g^{s_i}$ and a proof of knowledge of s_i.

Secure Two-Party Computation from Homomorphic Encryption

In order to evaluate functions securely, withstanding active attacks, zero-knowledge proofs are needed. The most important proofs are mentioned here for the discrete log setting. Known is Schnorr's protocol for proving the knowledge of x given the public value g^x and the extension of Okamoto proving knowledge of x, y given the common value $g^x h^y$. Furthermore there is the Chaum-Pedersen protocol for proving knowledge of x given the common value $(a,b) = (g^x, h^y)$ (note that this gives a proof for proving that a cipher text (a,b) is an encryption of zero). Applying OR-composition, one can distill from above

mentioned proofs a proof for proving that a cipher text (a,b) is an encryption of a bit. We note that those proofs can be simulated and hence do not leak any information about the input to the protocol except what is revealed by the protocol itself.

In order to be able to securely evaluate any function using an additive homomorphic encryption scheme, a protocol for secure multiplication is needed. Such protocols are described in "Berry Schoenmakers and Pim Tuyls, "Practical two-party computation based on the conditional gate", Advances in Cryptology – ASIACRYPT 2004, volume 3329 of Lecture Notes in Computer Science, pages 119-136, Springer-Verlag, Berlin, Germany, 2004", hereinafter [ST04]. [ST04] describes a suitable private multiplier gate and the conditional gate. Those protocols are simulatable even in the malicious case and guarantee to both users that the computations have been performed correctly.

First consider the situation where the encryptions $[x]=(a,b)=(g^r,g^x h^r)$ and $[y]=(c,d)$ are given with player P_1 knowing r,x. Player P_1 computes on its own a randomized encryption $[x,y]=(e,f)=(g^s,h^s)[y]^x$, with $s\in_R Z_q$, using the homomorphic properties. Finally player P_1 broadcasts $[xy]$ along with a proof showing that this is the correct output, which means that she proves knowledge of witnesses $r,s,x\in Z_q$ satisfying $a=g^r, b=g^x h^r, e=g^s c^x, f=h^s d^x$.

Next, we consider a multiplication gate is considered taking only encrypted inputs and for which the multiplier x is from a dichotomous (two-valued) domain, whereas the multiplicand y is unrestricted. It was called conditional gate in [ST04]. We present an explicit implementation for the two-party case. For sake of simplicity the formulation is done for the dichotomous domain $\{-1,1\}$. Domain $\{0,1\}$ or any other domain $\{a,b\}, a\neq b$, , can be used instead, as these domains can be transformed into each other by linear transformations: $x\rightarrow a'+(b'-a')(x-a)/(b-a)$ maps $\{a,b\}$ to $\{a',b'\}$. These transformations can be applied directly to homomorphic encryptions, transforming $[x]$ with $x\in\{a,b\}$ into $[x']$ with $x'\in\{a',b'\}$.

Let $[x],[y]$ denote encryptions, with $x\in\{-1,1\}\subseteq Z_q$ and $y\in Z_q$. The following protocol enables players P_1 and P_2, to compute an encryption $[xy]$ securely.

Player P_1 broadcasts an encryption $<< s_1 >>$, with $s_1 \in_R \{-1,1\}$. Then P_1 applies the private-multiplier multiplication protocol to multiplier s_1 and multiplicands $[x]$ and $[y]$, yielding random encryptions $[s_1 x]$ and $[s_1 y]$. Analogously, player P_2 broadcasts an encryption $<< s_2 >>$, with $s_2 \in_R \{-1,1\}$. Then P_2 applies the private-multiplier multiplication protocol to multiplier s_2 and multiplicands $[s_1 x]$ and $[s_1 y]$, yielding random encryptions $[s_1 s_2 x]$ and $[s_1 s_2 y]$.

The players jointly decrypt $[s_1 s_2 x]$ to obtain $s_1 s_2 x$. If decryption fails because $s_1 s_2 x \notin \{-1,1\}$, the protocol is aborted.

- Given $s_1 s_2 x$ and $[s_1 s_2 y]$, an encryption $\left[(s_1)^2 (s_2)^2 (xy)\right] = [xy]$ is computed publicly.

Note that this protocol takes only one threshold decryption, has a communication complexity of $26k$ bits (in total) and round complexity of two rounds. The evaluation of this gate requires 15 exponentiations per party while the private multiplier gate requires only 6 exponentiations.

Evaluation of the gates

AND with Encrypted Inputs

In order to evaluate an AND gate securely with two players given two encrypted bits $[x]$ and $[y]$, it suffices to run a conditional gate on those two inputs to compute $[xy]$.

AND Gate with one Encrypted and one Unencrypted Input

The secure evaluation of an AND gate where one player has a private bit x as input (unencrypted input) and an encrypted bit $[y]$ is available to both players is done by executing the private multiplier gate on the inputs x and $[y]$.

OR Gate with Encrypted Inputs

Given two encrypted bits $[x]$ and $[y]$, $[\mathrm{OR}(x, y)]$ is securely computed as follows.

The players run a conditional gate on the inputs $[x]$ and $[y]$ and obtain the result $[xy]$.

Using the homomorphic properties of the cryptosystem, they compute $[\mathrm{OR}(x,y)]$ as $[x+y-xy]$.

OR with an Encrypted and an Unencrypted Input

In order to perform a secure computation of an OR gate when one player has as private input a bit x (unencrypted input) and $[y]$ is given as public input, the players execute a private multiplier gate on x and $[y]$, yielding $[xy]$. Using the homomorphic properties of the encryption system the players compute $[\mathrm{OR}(x,y)]=[x+y-xy]$.

NOT Gate on Encrypted Inputs

The computation of $[\mathrm{NOT}(x)]$ given $[x]$ is done by computing $[1-x]$ publicly by using the homomorphic properties of the encryption system.

Applications

The described protocol is suitable for negotiation in general and more specifically for negotiating privacy preferences. It can be used in various settings where it is not sufficient to simply match attributes, but to allow users to negotiate a match based on more complicated preferences. Some examples are:

Non-humiliating dating.

The problem addressed in non-humiliating dating of humans is to determine whether two people share common interests in order to find out if the parties are a potential date (in its simplest form, a party's input is simply whether he or her is interested in the other party). To prevent humiliation, no interests are revealed unless shared by the other person, if any at all. In its simplest form, every party has a constant set of attributes. A more complex setting is where each party has several sets of attributes, corresponding to the different offers the party is willing to make, and a match has to be found without revealing any information about the policies outside the final match. One example application is that a person may at the same time have a different set of preferences for quick affairs and for a more permanent relationship. Furthermore, he does not want a partner matching the "permanent relationship set" to know that he was also looking for a quick affair, even if (or especially if) a match happens.

Business negotiations.

The protocol may further be used for classical negotiation deals, i.e., for buying goods or services. Classical private negotiation systems are one-dimensional, i.e.,

both parties define an amount of money they are willing to spent or want to get, respectively, and the system tells them if a deal can happen (and potentially, for how much). However, most negotiations today have more facets. The seller may offer some discount if paid in cash, or if several items are bought, and the buyer may pay more if home delivery is ensured, or the warranty is extended. Assuming the number of options is not exceedingly high, the protocol according to the invention delivers a practical way to privately negotiate the proper conditions.

Patients privacy. A pharmacist can test whether any conditions for a patient not to take a certain drug apply, without learning anything about the patients state. The conditions in this case can be rather complex - for example, a drug may not be prescribed to patients with a weak heart, or an HIV infection and a certain allergy, or under various sets of other medications.

It will be appreciated that the invention also extends to computer programs, particularly computer programs on or in a carrier, adapted for putting the invention into practice. The program may be in the form of source code, object code, a code intermediate source and object code such as partially compiled form, or in any other form suitable for use in the implementation of the method according to the invention. The carrier be any entity or device capable of carrying the program. For example, the carrier may include a storage medium, such as a ROM, for example a CD ROM or a semiconductor ROM, or a magnetic recording medium, for example a floppy disc or hard disk. Further the carrier may be a transmissible carrier such as an electrical or optical signal, which may be conveyed via electrical or optical cable or by radio or other means. When the program is embodied in such a signal, the carrier may be constituted by such cable or other device or means. Alternatively, the carrier may be an integrated circuit in which the program is embedded, the integrated circuit being adapted for performing, or for use in the performance of, the relevant method.

It should be noted that the above-mentioned embodiments illustrate rather than limit the invention, and that those skilled in the art will be able to design many alternative embodiments without departing from the scope of the appended claims. In the claims, any reference signs placed between parentheses shall not be construed as limiting the claim. Use of the verb "comprise" and its conjugations does not exclude the presence of elements or steps other than those stated in a claim. The article "a" or "an" preceding an element does not exclude the presence of a plurality of such elements. The invention may be implemented by means of hardware comprising several distinct elements, and by means of a suitably programmed computer. In the device claim enumerating several means, several of these

means may be embodied by one and the same item of hardware. The mere fact that certain measures are recited in mutually different dependent claims does not indicate that a combination of these measures cannot be used to advantage.

CLAIMS:

1. A method of automatic agreeing on a set of attributes between a plurality of stations in a computer network; the method including:

in each station determining at least one respective set of attributes that is acceptable for the station to be agreed with the other station by for each set of attributes selecting the attributes from a predetermined list of attributes and arranging the sets of attributes in a predetermined order;

using a predetermined multi-party computation protocol to evaluate in each station the respective ordered lists of acceptable sets without revealing to the other station any information other than a willingness to accept the agreed set, and to determine the agreed set based on the evaluation.

2. The method as claimed in claim 1, wherein the multi-party computation protocol is based on a homomorphic encryption system.

3. The method as claimed in claim 2, wherein the homomorphic encryption system is an El Gamal or Paillier type of encryption.

4. The method as claimed in claim 1, including the step of representing for each stations the acceptable sets of attributes using respective truth tables including a preference column that for each possible set of attributes of the predetermined list of attributes includes information on a level of acceptability of the set.

5. A method as claimed in claim 4, wherein the method includes:

selecting for a first one of the stations a generating subset of acceptable sets of attributes where an acceptable set of attributes of the first one of the stations is a generating subset if there is no other acceptable set of attributes that fully includes this acceptable set of attributes; and

selecting for a second one of the station a generating subset of acceptable sets of attributes where an acceptable set of attributes of the second one of the stations is a

generating subset if there is no other acceptable set of attributes that includes not at least one of the attributes of the acceptable set of attributes.

6. The method as claimed in claim 5, wherein the step of determining an agreed set of attributes includes forming a Boolean circuit implementing a process of matching the generating subsets, where the generating subsets and the respective preference columns for the generating subsets are used as input and an index representing a mutually acceptable set of attributes is generated as output.

7. The method as claimed in claim 6, including using the predetermined multi-party computation protocol to evaluate the Boolean circuit.

8. A method as claimed in claim 1, wherein the method includes using a predetermined selection criterion to select the agreed set among a plurality of mutually acceptable sets of attributes.

9. A method as claimed in claim 1, wherein the method further includes agreeing on a set of obligations between the stations; each of the stations being associated with at least one respective set of obligations selected from a predetermined list of obligations; the method including using the multi-party computation protocol to determine the agreed set of obligations without revealing other information than a willingness to accept the agreed set of obligations.

10. A computer program product for causing a processor to execute the method as claimed in claim 1.

11. A system for automatic agreeing on a set of attributes; the system including a plurality of stations arranged to communicate through a computer network; each station including:

respective means for determining at least one respective set of attributes that is acceptable for the station to be agreed with the other station by for each set of attributes selecting the attributes from a predetermined list of attributes and arranging the sets of attributes in a predetermined order; and

respective means for executing a predetermined multi-party computation

protocol to evaluate in each station the respective ordered lists of acceptable sets without revealing to the other station any information other than a willingness to accept the agreed set, and to determine the agreed set based on the evaluation.

12. A station for automatic agreeing on a set of attributes with at least one other station and arranged to communicate through a computer network with the at least one other station, including:

means for determining at least one set of attributes that is acceptable for the station to be agreed with the at least one other station by for each set of attributes selecting the attributes from a predetermined list of attributes and arranging the sets of attributes in a predetermined order; and

means for participating in a predetermined multi-party computation protocol to evaluate the ordered list of acceptable sets without revealing to the at least one other station any information other than a willingness to accept the agreed set, and to determine the agreed set based on the evaluation.

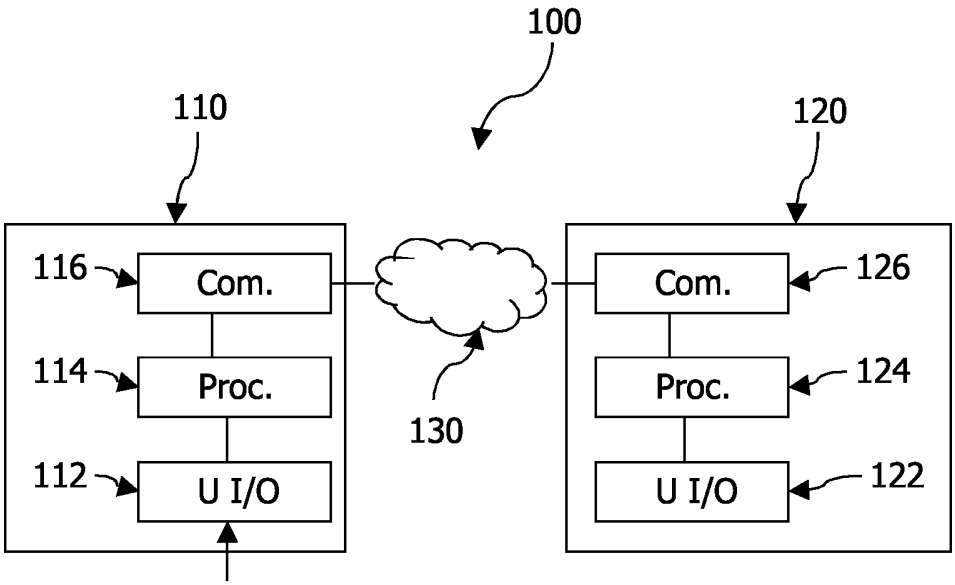

100
110
120
116
Com.
126
Com.
114
Proc.
124
Proc.
130
112
U I/O
122
U I/O

FIG.1

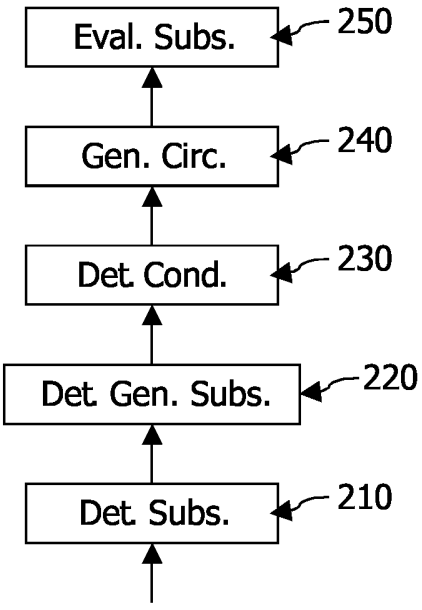
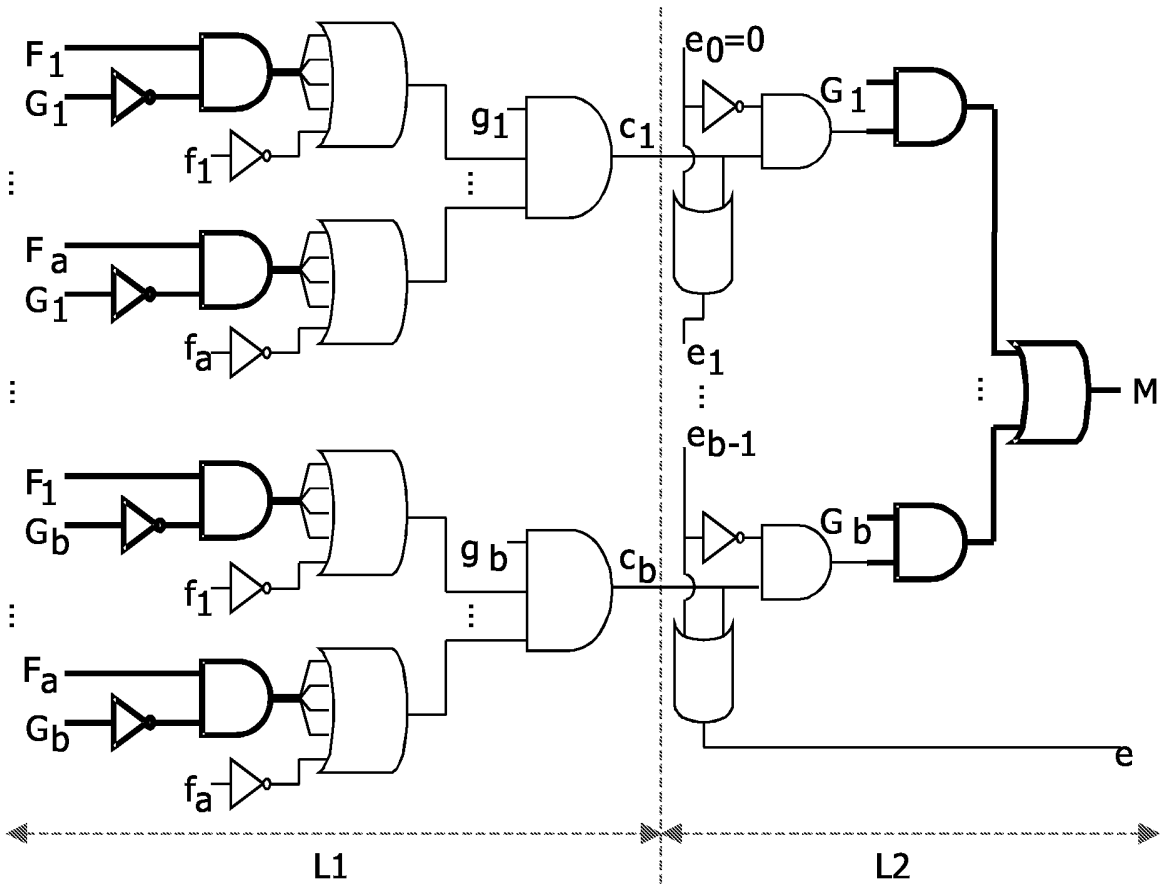

Eval. Subs.
250
Gen. Circ.
240
Det. Cond.
230
Det. Gen. Subs.
220
Det. Subs.
210

FIG.2



Legend:

AND
OR
NOT
bitwise AND
bitwise OR
bitwise NOT
single wire
bus
split bus into single wires
repeat single wire on all wires of bus

FIG.3

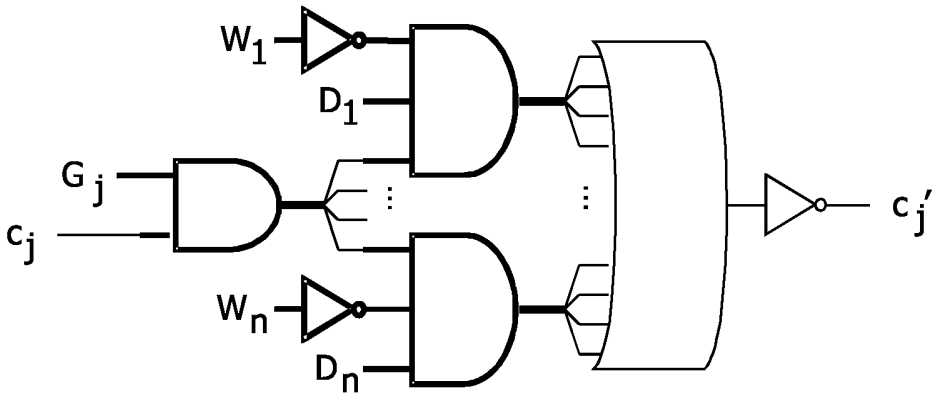


FIG.4A

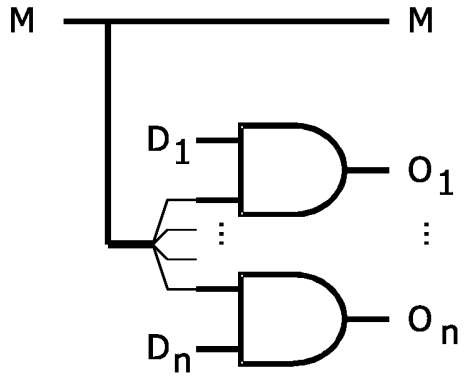


FIG.4B