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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
16 February 2017 (16.02.2017)

WIPO | PCT

(10) International Publication Number
WO 2017/025597 A1

(51) **International Patent Classification**:
H04L 9/08 (2006.01)

(21) **International Application Number**:
PCT/EP2016/069133

(22) **International Filing Date**:
11 August 2016 (11.08.2016)

(25) **Filing Language**: English

(26) **Publication Language**: English

(30) **Priority Data**:
15180484.6 11 August 2015 (11.08.2015) EP

(71) **Applicant**: **KONINKLIJKE PHILIPS N.V.** [NL/NL]; High Tech Campus 5, 5656 AE Eindhoven (NL).

(72) **Inventors**: **GUTIERREZ, Jaime**; c/o High Tech Campus 5, 5656 AE Eindhoven (NL). **GOMEZ, Domingo**; c/o High Tech Campus 5, 5656 AE Eindhoven (NL). **TOLHUIZEN, Ludovicus Marinus Gerardus Maria**; c/o High Tech Campus 5, 5656 AE Eindhoven (NL). **RIETMAN, Ronald**; c/o High Tech Campus 5, 5656 AE Eindhoven (NL). **GARCIA MORCHON, Oscar**; c/o High Tech Campus 5, 5656 AE Eindhoven (NL).

(74) **Agents**: **COOPS, Peter** et al.; High Tech Campus, Building 5, 5656 AE Eindhoven (NL).

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

(84) **Designated States** *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

[Continued on next page]

(54) **Title**: KEY SHARING DEVICE AND METHOD

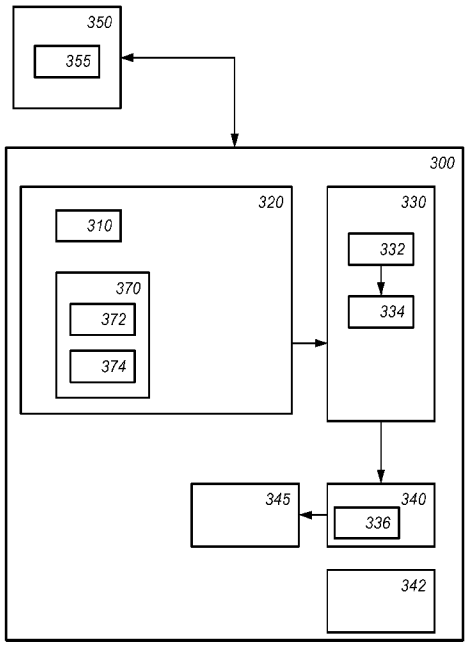

Fig. 2a

(57) **Abstract**: A network device (300) arranged to determine a shared key with a second network device (350) is presented. The first network device is arranged to substitute an identity number of the second network device into a univariate private key polynomial to obtain an intermediate key. The network device comprises - a key derivation unit arranged to - extract from the intermediate key multiple bit-strings, the multiple bit-strings being non-overlapping, consecutive substrings of the intermediate key, each two adjacent bit-strings of the multiple bit-strings are separated in the intermediate key by at least one bit which does not belong to any of the multiple bit-strings, and - derive the shared key from the multiple bit-strings.

WO 2017/025597 A1

Published:

— *with international search report (Art. 21(3))*

Key sharing device and method

FIELD OF THE INVENTION

The invention relates to a network device, a key material generation device, a key sharing method, a key material generation method, a computer program, and a computer readable medium.

BACKGROUND

In cryptography, a key-agreement protocol is a protocol whereby two or more parties that may not yet share a common key can agree on such a key. Preferably, both parties can influence the outcome so that neither party can force the choice of key. An attacker who eavesdrops on all communication between the two parties should learn nothing about the key. Yet, while the attacker who sees the same communication learns nothing or little, the parties themselves can derive a shared key.

Key agreement protocols are useful, e.g., to secure communication, e.g., to encrypt and/or authenticate messages between the parties.

Practical key agreements protocols were introduced in 1976 when Whitfield Diffie and Martin Hellman introduced the notion of public-key cryptography. They proposed a system for key agreement between two parties which makes use of the apparent difficulty of computing logarithms over a finite field GF(q) with q elements. Using the system, two users can agree on a symmetric key. The symmetric key may then be used for say, encrypted communication between the two parties.

The Diffie-Hellman system for key agreement is applicable when the parties do not yet have a shared secret. The Diffie-Hellman key agreement method requires resource-heavy mathematical operations, such as performing exponentiation operations over a finite field. Both the exponent and the field size may be large. This makes key agreement protocols less suitable for low-resource devices. On the other hand key agreement protocols would be very useful in resource-restrained devices. For example, in application areas such as the internet of things, ad-hoc wireless networks, and the like, key agreement could be used to protect links between devices. Another example is communication between a reader and an

electronic tag, say a card reader and a smart card, or a tag reader and tag, e.g., an RFID tag or an NFC tag.

Another approach to the problem of setting up secure connections between pairs of network devices in a given communications network is given in C. Blundo, A. De Santis, A. Herzberg, S. Kutten, U. Vaccaro and M. Yung, "Perfectly-Secure Key distribution for Dynamic Conferences", Springer Lecture Notes in Mathematics, Vol. 740, pp. 471-486, 1993 (referred to as 'Blundo').

This system assumes a key material generation device, e.g., central authority, also referred to as the network authority or as the Trusted Third Party (TTP), that generates a bivariate polynomials f(x,y), with coefficients in the finite field F with p elements, wherein p is a prime number or a power of a prime number. Each device has an identity number in F and is provided with local key material by the TTP. For a device with identifier number η, the local key material is the coefficients of the polynomial $f(\eta,y)$. If a device η wishes to communicate with device η', it uses its key material to generate the key $K(\eta, \eta') = f(\eta, \eta')$. As f is symmetric, the same key is generated. The local key material is secret. Knowledge of the local key material would directly compromise the system. In particular it would allow an eavesdropper to obtain the same shared key. The method requires that each device in a network of devices has its own unique identity number and local key material.

A problem of this key sharing scheme occurs if an attacker knows the key material of t+1 or more devices, wherein t is the degree of the bivariate polynomial. The attacker can then reconstruct the polynomial f(x,y). At that moment the security of the system is completely broken. Given the identity numbers of any two devices, the attacker can reconstruct the key shared between this pair of devices.

International Patent Application PCT/EP2013/056730 by the same applicant, published as WO/2013/174554, discloses a key sharing technique that resists obtaining the root key material from the key material of hacked devices (also called colluding devices). Unlike the Blundo scheme this key sharing agreement system starts with multiple bivariate polynomials. Univariate polynomials obtained from the bivariate polynomials are added over different moduli, making reconstruction of the root key material harder for the attacker.

SUMMARY OF THE INVENTION

There is a need to further complicate the complexity of the reconstructing the root key material from the private key material of multiple colluding (e.g. hacked) devices;

also known as collusion attacks. It would be advantageous to have an improved system for sharing keys among devices.

A first network device arranged to determine a shared key with a second network device is provided. The first network device comprises

- an electronic storage storing a univariate private key polynomial and a public global reduction integer obtained from an external key material generation device arranged to configure at least the first and second network device for key sharing, the storage further storing a first identity number for the first network device used by the key material generation device to generate the univariate private key polynomial,
- a communication unit arranged to obtain a second identity number of the second network device, the second network device being different from the first network device,
- a polynomial manipulation unit arranged to
  - substitute the second identity number into the univariate private key polynomial,
  - reduce the result of the substituting modulo the public global reduction integer thus obtaining an intermediate key, and
- a key derivation unit arranged to
  - extract from the intermediate key multiple bit-strings, the multiple bit-strings being non-overlapping, consecutive substrings of the intermediate key, each two adjacent bit-strings of the multiple bit-strings are separated in the intermediate key by at least one bit which does not belong to any of the multiple bit-strings, and
  - derive the shared key from the multiple bit-strings.

The first network device has a higher resistance against collusion attacks. A lattice built from the univariate polynomials of colluding devices has much larger coefficients than a comparable lattice which does not use spacing between adjacent bit strings. Furthermore, the spacing complicates the lattice; for example, one way to deal with the absence of information on the intermediate key in the area where spacing is used, is to model each one of the multiple bit strings separately in the lattice, which directly increases its dimension. Thus, if the degree of the bivariate polynomials is lowered, but spacing is introduced, the dimension of the resulting lattice may be comparable.

If the univariate private key polynomial were obtained by summing polynomials reduced over different moduli, the shared keys are on the one hand more resilient against collusion attacks, but on the other hand there is a chance that the shared key

derived in a first network device is not exactly the same as the shared key derived in a second network device. If needed, this problem can be resolved by exchanging key-reconciliation data. Only one of the first and second devices needs to do reconciliation. In an embodiment, a first network device initiates the key sharing and send his identity number to a second network device, the second network device responds to the key sharing request by sending reconciliation data and its identity number; allowing the first network device to arrive at the same shared key as the second network device.

In an embodiment, the first device comprising a key-reconciliation unit arranged to compute key-reconciliation data from the multiple bit-strings, the communication unit being further arranged to send the key-reconciliation data to the second device.

In an embodiment, the communication unit is further arranged to receive key-reconciliation data from the second device, the first device comprising a key-reconciliation unit arranged to modify the multiple bit-strings to conform to the received key-reconciliation data, the shared key being derived from the modified multiple bit-strings.

The univariate private key polynomial may be generated by a key material generation device. In a first embodiment, the key material generation device uses a single bivariate polynomial without private reduction integers. Univariate private key polynomials that are obtained in this way have the property that they do not need reconciliation. They are however more vulnerable to collusion attacks.

In a second embodiment the key material generation device obtains a set of univariate polynomials by

- for each particular polynomial of a first private set, substituting the identity number into said particular polynomial f_i and reducing modulo the reduction integer associated with said particular polynomial, and
- summing the set of univariate polynomials and reducing modulo the public global reduction integer.

Summing polynomials reduced over different private reduction integers is an unusual operation. As the modulo operations are not compatible with each other, the reconciliation becomes necessary. On the other hand, mathematical analysis of the univariate polynomials has become increasingly complicated. Applying spacing in the intermediate key appears to be especially beneficial when the univariate polynomials are derived from multiple bivariate polynomials in this way instead of from a single bivariate polynomial.

In an embodiment, the amount of spacing between the multiple-bit strings is linked to the degree of the bivariate polynomials used to generate the univariate private key

polynomials. In an embodiment, the spacing number of bits s has $(\alpha + 1)B$ bits or more, wherein

- α is the highest degree in a single variable of the bivariate polynomials in the first private set
- the first and second identity numbers have an identity number length B bits. Having such spacing guarantees that keys of first and second device are quite close.

In a further aspect of the invention concerns a key material generation device arranged to configure a first network device for sharing a shared key, and a method of key sharing and key material generation.

The network devices described herein may be applied in a wide range of practical applications. Such practical applications include: lighting networks, sensor networks, home automation, HVAC (heating, ventilation, and air conditioning) networks, industry networks, control networks, ad-hoc wireless communication networks, etc. For example, lighting nodes may include a network device.

A method according to the invention may be implemented on a computer as a computer implemented method, or in dedicated hardware, or in a combination of both. Executable code for a method according to the invention may be stored on a computer program product. Examples of computer program products include memory devices, optical storage devices, integrated circuits, servers, online software, etc. Preferably, the computer program product comprises non-transitory program code means stored on a computer readable medium for performing a method according to the invention when said program product is executed on a computer.

In a preferred embodiment, the computer program comprises computer program code means adapted to perform all the steps of a method according to the invention when the computer program is run on a computer. Preferably, the computer program is embodied on a computer readable medium.

Another aspect of the invention provides a method of making the computer program available for downloading. This aspect is used when the computer program is uploaded into, e.g., Apple's App Store, Google's Play Store, or Microsoft's Windows Store, and when the computer program is available for downloading from such a store.

BRIEF DESCRIPTION OF THE DRAWINGS

Further details, aspects, and embodiments of the invention will be described, by way of example only, with reference to the drawings. Elements in the figures are

illustrated for simplicity and clarity and have not necessarily been drawn to scale. In the Figures, elements which correspond to elements already described may have the same reference numerals. In the drawings,

Figure 1 schematically shows an example of an embodiment of a key material generation device,

Figure 2a schematically shows an example of an embodiment of a first network device,

Figure 2b schematically shows an example of an embodiment of an intermediate key,

Figure 3a is a schematic block diagram of a key sharing system 100,

Figure 3b is a schematic block diagram of a key sharing system 102

Figure 4a is schematic block diagram of an integrated circuit 400,

Figure 4b schematically shows a computer readable medium (1000) having a writable part (1020) comprising a computer program according to an embodiment,

Figure 5 schematically shows a flowchart illustrating a key sharing method,

Figure 6 schematically shows a flowchart illustrating a key material generation method.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

While this invention is susceptible of embodiment in many different forms, there are shown in the drawings and will herein be described in detail one or more specific embodiments, with the understanding that the present disclosure is to be considered as exemplary of the principles of the invention and not intended to limit the invention to the specific embodiments shown and described.

In the following, for the sake of understanding, elements of embodiments are described in operation. However, it will be apparent that the respective elements are arranged to perform the functions being described as performed by them.

Further, the invention is not limited to the embodiments, and the invention lies in each and every novel feature or combination of features described above or recited in mutually different dependent claims.

Below an embodiment of the key sharing method is described in mathematical terms. The key sharing method may be implemented in devices as described below, e.g., on a key material generation device (200), a network device (300), in a key sharing system (100), (102) and the like.

In the embodiment below network devices are configured to obtain a shared key. Generally, the shared key will have fewer bits than the identity numbers of the network devices. Multiple of such shared keys may be combined to obtain a larger key, but this is not necessary. The method has a set-up phase and a use phase. The set-up phase may include initiation steps and registration steps. The initiation steps do not involve the network devices.

The initiation steps select system parameters. The initiation steps may be performed by the trusted third party (TTP). The system parameters may also be regarded as given inputs. In that case the trusted third party need not generate them, and the initiation steps may be skipped. For example, the trusted third party may receive the system parameters from a device manufacturer. The device manufacturer may have performed the initiation steps to obtain the system parameters. For convenience of exposition we will refer to the trusted third party as performing the initiation steps, bearing in mind that this is not necessary.

Initiation steps

The desired key length for the key that will be shared between devices in the use phase is selected; this key length is referred to as 'b'. The shared key will be selected from different locations in an intermediate key. The number of multiple bit-strings is referred to as t. The bit length of the multiple bit-strings will be referred to as $b_1, \ldots, b_t$; the bit-lengths are given in the order in which the bit-strings appear in the intermediate key, starting with b_1 which is closest to the LSB of the intermediate key. The parameters t and $b_1, \ldots, b_t$ are selected in the initiation phase. Obtaining a shared key from the multiple bit-strings is further explained below. We have $t > 1$ and $b = \sum_{i=1}^{t} b_i$. In an embodiment, $t = 2$.

In an embodiment, two adjacent bit-strings of the multiple bit-strings are separated in the intermediate key by a same spacing number of bits (s). The total number of bits in the intermediate key used for spacing, and not for the shared key may be referred to as S. In an embodiment, $S = ts$.

The desired identity number length is also selected. During the later registration steps each device will be associated with an identity number of identity number length; the identity number length is referred to as 'B'. The length of numbers are measured in bits. In an embodiment $b \leq B$.

It has been found that $b < B$, increases resilience to so-called collusion attacks. In a collusion attack, an attacker obtains information on the shared key used between a target network node and multiple colluding network nodes. The amount of information learned

from each additional colluding network node is of size b. However, the amount of information that needs to be reconstructed in order to break commutation between the target network node and non-colluding network nodes grows with B. In an embodiment B is a multiple of b; say B is at least $2b$, or for recommended security levels, B is at least $4b$. One may also have that $b = B$. For example, $b = B = 64$, or $b = B = 128$. Having $b = B$ reduces key storage. It is noted that the present way of selecting a shared key also improves resistance against collusion attacks, so that having $b = B$ is acceptable in more applications.

Next the parameters are selected. The desired degree is selected; the degree controls the degree of certain polynomials. The degree will be referred to as 'α', it is at least 1. Although the system will work with $\alpha = 1$, the underlying problem changes in nature from α>1. We will assume $\alpha \geq 2$ from here on. A more secure application may use a higher value of α, say 3 or 4, or even higher; for example, in embodiments m may even be 10 or more.

For a simple application also α =1 is possible. The case $\alpha = 1$ is related to the so called 'hidden number problem'; higher "α" values are related to the extended hidden number problem confirming that these cases are hard to break. The value $\alpha = 1$, although possible, is not recommended, and should only be considered for very low security applications. For low security application a value of $\alpha > 2$, say $\alpha = 3$ is possible. However, for high security $\alpha \geq 32$ is recommended, say $\alpha = 32$.

The number of polynomials is selected. The number of bivariate polynomials will be referred to as 'm'. A practical choice for m is 2. A more secure application may use a higher value of m, say 3 or 4, or even higher.

Note that a low-complexity application, say for resource bounded devices may use $m = 1$. The value $m = 1$, although possible, is not recommended, and should only be considered for low security applications. Higher values of security parameters α and m increase the complexity of the system and accordingly increase its intractability. More complicated systems are harder to analyze and thus more resistant to cryptanalysis. Below it is assumed that $m \geq 2$.

A public modulus N is selected satisfying $2^{S+b-1} \leq N$. Preferably, public modulus N is chosen to have exactly $S + b$ bits, and thus that also $N < 2^{S+b}$. For example, N may be chosen at random in this interval. In an embodiment, the spacing is equal for each one of the multiple bit-strings, in this case $S = ts$. In an embodiment, N is odd. In particular, in a practical embodiment N is odd and has exactly $ts + b$ bits. In an embodiment, $s \geq (\alpha + 1)B$. In particular $s = (\alpha + 1)B$. This choice for s allows less reconciliation. Constant spacing,

including at the MSB, with $s = (\alpha + 1)B$ and $S = ts$ is assumed from now on unless otherwise indicated.

Often the degree α, number of polynomials m, key length b, number of bit-strings t, spacing s, sizes of key parts b_i, will be pre-determined, e.g., by a system designer and provided to the trusted party as inputs. The public modulus may also be fixed, say in a standard, but more typically will be selected by a key material generation device during generation of the parameters.

A number of m private moduli $p_1, p_2, \ldots, p_m$ are selected. Moduli are positive integers. Each selected number satisfies a relationship with the public global modulus N. In an embodiment, each private modulus satisfies the following relationship with public global modulus N.

$$p_i = N - \sum_{k=1}^{t} \beta_i^{(k)} 2^{s(k-1)+\sum_{\ell=1}^{k} b_\ell}$$

for some integers $\beta_i^{(k)}$ with $\beta_i^{(k)} \leq 2^B$ for $1 \leq k \leq t$. For example, the $\beta_i^{(k)}$ may be random B-bits integers; more preferably they have exactly B bits, i.e., $2^{B-1} \leq \beta_j < 2^B$. In the exponent of 2 in the above formula, $s(k-1)$, equals the amount of spacing introduced up to the introduction of $\beta_i^{(k)}$. There is more freedom in choosing the private moduli. Embodiment may use different choices for p_i, or if using the above construction, $\beta_i^{(k)}$. However, the above construction is both convenient in implementations, and gives good guarantees for the amount of reconciliation (see below).

For $m > 1$, the system is more complicated, and thus more secure, since modulo operation for different moduli are combined even though such operations are not compatible in the usual mathematical sense. For this reason it is advantageous to choose the selected private moduli p_j as pairwise distinct.

A number of m bivariate polynomials $f_1, f_2, \ldots, f_m$ of degrees α_j are generated. Preferably, the bivariate polynomials are symmetric; this allows all network devices to agree on a shared key with each other network device. These bivariate polynomials may also be chosen asymmetric. All degrees satisfy $\alpha_j \leq \alpha$, and for at least one j, we have $\alpha_j = \alpha$. A better choice is to take each polynomial of degree a. A bivariate polynomial is a polynomial in two variables. A symmetric polynomial f satisfies $f(x, y) = f(y, x)$. Each polynomial f_j is evaluated in the finite ring formed by the integers modulo p_j, obtained by computing modulo p_j. The integers modulo p_j form a finite ring with p_j elements. The coefficients of polynomial f_j are integers, and represent an element in the finite ring defined by modulo p_j operations. In

an embodiment the polynomial f_j is represented with coefficients from 0 up to $p_j - 1$. The bivariate polynomials may be selected at random, e.g., by selecting random coefficients within these bounds.

The security of the key sharing depends on these bivariate polynomials as they are the root key material of the system; so preferably strong measures are taken to protect them, e.g., control procedures, tamper-resistant devices, and the like. Preferably the selected integers $p_1, p_2, \ldots, p_m$ are also kept secret, including the values $\beta_j^{(k)}$ corresponding to p_j, though this is less critical. We will refer to the bivariate polynomials also in the following form: for j=1, 2,..., m, we write $f_j(x, y) = \sum_{i=0}^{\alpha} f_{i,j}(x) y^i$.

The above embodiment can be varied in a number of ways. The restrictions on the public and private moduli may be chosen in a variety of ways, such that obfuscation of the univariate polynomial is possible, yet that the shared keys obtained at network devices remain sufficiently close to each other sufficiently often. What is sufficient will depend on the application, the required security level, and the computing resources available at the network devices. The above embodiment combines positive integers such that the modular operations which are carried out when generating the polynomials shares are combined in a non-linear manner when they are added over the integers, creating a non-linear structure for the local key material stored on a network device. The above choice for N and p_j has the property that: (i) the size of N is fixed for all network devices and linked to α; (ii) the non-linear effect appears in the coefficients forming the key material stored on the device.

Registration steps

In the registration step each network device is assigned key material (KM). The key material is unique to a network device.

A network device is associated with an identity number A. The identity number may be assigned on demand, e.g. by the TTP, or may already be stored in the device, e.g., stored in the device at manufacture, etc. The bit size of A is B bits. Generating A may be done in a variety of ways. For high security the low bits of A are random. For example, A may be selected as a random number; A may be the hash of a further identity number, say a serial number, possibly truncated to B bits.

The TTP generates a set of key material for a device A as follows:

$$KM^A(x) = \sum_{j=1}^{m} < f_j(x, A) >_{p_j} = \sum_i C_i^A x^i$$

It is possible to add further obfuscating numbers to this, as follows:

$$KM^A(x) = \sum_{j=1}^{m} < f_j(x,A) >_{p_j} + 2^b \sum_{i=0}^{\alpha} \epsilon_{A,i} X^i = \sum_i C_i^A x^i$$

Wherein $KM^A(x)$ is the key material of a device with identity number A; x is a formal variable. Note that the key material is non-linear in the identity number A. The notation $< \cdots >_{p_j}$ denotes reducing modulo p_j each coefficient of the polynomial between the brackets. Stated differently, we have that $C_i^A = \sum_{j=1}^{m} \langle f_{i,j}(A) \rangle_{p_j} + 2^b \epsilon_{A,i}$. This additional obfuscation is optional.

The notation '$\epsilon_{A,i}$' denotes a random integer, which is an example of an obfuscating number, such that $|\epsilon_{A,i}| < 2^{(\alpha+1-i)b}$. Note that any one of the random integers may be positive or negative. The random numbers ϵ are generated again for each device. The term $\sum_{i=0}^{a} \epsilon_{A,i} X^i$ thus represents a polynomial in X of degree a, of which the coefficient length is shorter with increasing degree. Alternatively, a more general, but more complicated condition is that $\sum_{i=0}^{a} |\epsilon_{A,i}| \cdot 2^{ib}$ is small, e.g., $< 2^{a+1}$. The mixing effect over different finite rings provides the largest contribution to security, the use of obfuscating numbers is thus optional.

All other additions may either use the natural integer arithmetic, i.e., in the ring $\mathbb{Z}$, or (preferably) they use addition modulo N. So the evaluation of the univariate polynomials $\sum_{j=1}^{m} < f_j(x,A) >_{p_j}$ is each individually done modulo a smaller modulus p_j but the summation of these reduced univariate polynomials themselves is preferably done modulo N. Also adding the obfuscating polynomial $2^b \sum_{i=0}^{a} \epsilon_{A,i} X^i$ may be done using natural integer arithmetic or, preferably, modulo N. The key material comprises the coefficients C_i^A with $i = 0, \ldots, a$. The key material may be presented as a polynomial as above. In practice, the key material may be stored as a list, e.g., an array, of the integers C_i^A. The device A also receives the numbers N and b_i and information of the spacing, e.g., s; so that A and B may extract the multiple bit strings in the same manner. Manipulation of polynomials may be implemented, e.g., as manipulation of arrays containing the coefficients, e.g., listing all coefficient in a predetermined order. Note that polynomials may be implemented, in other data structures, e.g., as an associative array (also known as a 'map') comprising a collection of (degree, coefficient) pairs, preferably such that each coefficient appears at most once in the collection. The coefficients C_i^A that are provided to the device may be in the range 0, 1, ..., N-1.

Univariate private key polynomials generated in this fashion have the property that they may be used to derive shared keys, have resistance against collusion attacks and control over the distance between the shared key derived at the respective devices (if any). These properties are a result of the process used to generate the univariate private polynomials.

Use phase

Once two devices have an identity number A and B and received the key material from the TTP, they may use their key material to obtain a shared key, shared between them. Device A may perform the following steps, to obtain his shared key. First, device A obtains the identity number B of device B. Then device A first generates the intermediate key by computing the following:

$$K(A,B) = < KM^A(x)|_{x=B} >_N = < \sum_i C_i^A B^i >_N$$

That is, A evaluates his key material, seen as an integer polynomial, for the identity number B; the result of evaluating the key material is an integer. Next device A reduces the result of the evaluation modulo the public modulus N. The angle brackets indicate a modulo operation. The intermediate key is the result of the modulo N operation.

Figure 2b schematically shows an example of an embodiment of an intermediate key 4000. Intermediate key 4000 is a bit string, the least significant (LSB) and most significant bits (MSB) being indicated in figure 2b.

From the intermediate key, multiple bit-strings are extracted. The multiple bit-strings are non-overlapping, consecutive substrings of the intermediate key. Each two adjacent bit-strings of the multiple bit-strings are separated in the intermediate key by at least one bit which does not belong to any of the multiple bit-strings. In figure 2b, three bit-strings are indicated: bit-strings 4012, 4022, and 4032. Bit strings 4012 and 4022 are separated by spacing 4020. Bit strings 4022 and 4032 are separated by spacing 4030.

In figure 2b, a spacing 4010, which includes the MSB of intermediate key 4000 separates bits string 4012 from the MSB part of the intermediate key. That is, a most significant part of the intermediate key is not included in the multiple bit strings. Not using the MSB part improves reconciliation.

In the embodiment shown in figure 2b, each two adjacent bit-strings of the multiple bit-strings are separated in the intermediate key by a same number of spacing bits: spacing number of bits (s). That is spacing parts 4010, 4020 and 4030 each have s bits. The multiple bit strings have lengths b_i. Also the multiple bit-strings may have equal lengths. As shown in figure 2b, the bit strings are counted starting from the LSB.

The multiple bit-strings may be obtained as follows:

The first bit-strings may be the least b_1 least significant bits of intermediate key $K(A,b)$, e.g.:

$$K_{A,B}^{(1)} = \langle K(A,B) \rangle_{2^{b_1}}$$

For $2 \leq k \leq t$, the k-th block $K_{A,B}^{(k)}$ may be b_k consecutive bits of $K(A,B)$. Bit-string k and bit-string $k-1$ are separated by s bits. In an embodiment $s = (\alpha+1)B$ bits. The k-th bit-string may be computed as:

$$K_{A,B}^{(k)} = \langle \left\lfloor \frac{K(A,B)}{2^{s(k-1)+\sum_{l=1}^{k-1} b_l}} \right\rfloor \rangle_{2^{b_k}}$$

The shared key may be derived from the multiple bit-strings. For example, they may be concatenated. The result will be referred to as A's shared key with B, it is an integer in the range of 0 up to $2^b - 1$. For its part, device B can generate B's shared key with A by evaluating its keyed material for identity A and reducing the result modulo N and extracting the same multi-bit strings. Device B will obtain multiple bit-strings $K_{B,A}^{(k)}$.

In the example, given above the first bit string includes the least significant bit of the intermediate key. This has the advantage that the intermediate key is shorter, and thus reduces computation time. On the other hand, the most significant bits are not included in a bit string. This has the advantage that shared keys are closer to each other between devices A and B, and thus shortens reconciliation; in an embodiment the s most significant bits of the intermediate key are excluded.

If the bivariate polynomials in the root key material are symmetric A's shared key with device B and device B's shared key with A are often, though not necessarily always, equal. The particular requirements on the integers $p_1, p_2, \ldots, p_m$, and on the random numbers ϵ are such that the keys are often equal and almost always close to each. If devices A and B have obtained the same shared key, then they may use it as a symmetric key which is shared between devices A and B; for example, it may be used for a variety of cryptographic applications, for example, they may exchange one or more messages encrypted and/or or authenticated using the shared key. Preferably, a key derivation algorithm is applied to the shared key for further protection of the master key, e.g., a hash function may be applied. Even if devices A and B have not obtained the same shared key, it is certain that these keys are close to each other. It can mathematically be shown that for $k = 1$

$$K_{A,B}^{(1)} \in \left\{ \langle K_{B,A}^{(1)} + jN \rangle_{2^{b_1}} \middle| -2m \leq j \leq 2m \right\},$$

and that for $2 \leq k \leq t$

$$K_{A,B}^{(k)} \in \left\{ \langle K_{B,A}^{(k)} + \left\lfloor \frac{jN}{2^{s(k-1)+\sum_{l=1}^{k-1} b_l}} \right\rfloor + e \rangle_{2^{b_k}} \middle| -2m \leq j \leq 2m \text{ and } -m-3 \leq e \leq m+3 \right\}$$

To ensure that the shared key is the same at both devices the devices may enter a so-called reconciliation phase. For example, device A may compute key-reconciliation data from the multiple bit-strings, and send it to device B. Or the other way round, device B may compute key-reconciliation data from the multiple bit-strings, and send it to device A.

The receiving party, say device A, may modify the multiple bit-strings so that they conform to the received key-reconciliation data, the shared key being derived from the modified multiple bit-string; e.g., concatenating the modified multiple bit-string, applying a key derivation function, hashing them etc. Alternatively, key reconciliation may comprise a number c of least significant bits of each of the multiple bit-strings. In an embodiment, $t \leq 4$, this reduces reconciliation; in particular t may be 2 to minimize reconciliation.

Key-reconciliation data may be a cryptographic hash over the multiple bit strings, e.g., a sha-1 hash over the concatenation of the multiple bits strings. Device A may vary his multiple bit-strings within the above identified parameters until key-reconciliation data computed over the modified bit-strings, e.g., the same hash function computed there over, equals the received key-reconciliation data.

For example, device A may generate all multiple bit-strings that conform to the above bounds until a set of multiple bit-strings is found that conforms to the received key reconciliation data. For example, device A may use a set of nested for-next loops; each loop generating the allowed values of one of the multiple-bit strings.

The selected m private moduli, $p_1, p_2, \ldots, p_m$, are preferably pairwise relatively prime. If these numbers are pairwise relatively prime the lack of compatibility between the modulo operations is increased. Obtaining pairwise relatively prime numbers may be obtained by selecting the integers in order, testing for each new integer if all pairs of different numbers are still relatively prime, if not the just selected number is removed from the set. This procedure continues until all m numbers are selected. The complexity increases even further by requiring that the selected m private moduli, $p_1, p_2, \ldots, p_m$ are distinct prime numbers.

Below some further examples are given, which may be used in embodiments. These numbers are examples, different choices may be made, as indicated herein.

In a first example, $B = b = 64$, $t = 2$, $b_1 = b_2 = 32$,

In a second example, $B = 128$, $b = 64$, $t = 2$, $b_1 = b_2 = 32$,

In a third example, $B = 128$, $b = 128$, $t = 4$, $b_1 = b_2 = b_3 = b_4 = 32$,

In all these examples, the root key material may be chosen with, say, $\alpha = 30$, and $m = 10$; Spacing may be chosen constant with $s = (\alpha + 1)B$ and size of N is $ts + b$. Size of b_i may be chosen larger or smaller than 32. There is a practical preference for power of two for the various parameters; From a cryptographical point of view there is however no need to restrict parameters to powers of two.

In the first example given above, the public global modulus N has length $2 * 31 * 64 + 64 = 63 * 64$ bits. The key material has $(\alpha + 1) = 31$ coefficients, each of $63 * 64$ bits, so in total $31 * 63 * 64$ bits.

We compare this first example with a comparative key sharing scheme which works the same except that t=1, i.e., in which the shared key is obtained from one consecutive bit-string instead of multiple strings. The only spacing, is the spacing between the MSB of the intermediate key and the single bit string. In the comparative system we have t=1, and $b_1 = b = B = 64$. To make a fair comparison we select a degree α' such that the amount of key material is similar. In the comparative example, we have $(\alpha' + 1)((\alpha' + 1)B + b)$, thus solving for $(\alpha' + 1)((\alpha' + 1)B + b) = 31 * 63 * 64$ gives α' =43. With this degree the public modulus in the comparative example has length $((\alpha' + 1)B + b) = 45 * 64$. So for the same number of key material bits, the comparative example has higher degree bivariate polynomials but no spacing.

The best known attacks on both these systems are lattice attacks. It turns out that the spacing complicates the lattice to such an extent that, although the degree of the bivariate polynomials is lower (30 versus 43) the dimension of the lattice dimension for both the first and comparative example is comparable. However, in the first example, the lattice uses a bit size of 63*64 bits, whereas in the comparative example, the bit size is only 45*64. In other words, by moving from a key-sharing example (the comparative example) without spacing to a key-sharing example with spacing, keeping the size of the shared key and the amount of key material constant, attacks require 40% larger lattices of similar dimension.

Another alternative to multiple bit-strings from the same intermediate key is to create multiple parallel key agreements so that multiple bit-strings may be obtained from multiple intermediate keys. The parallel system is easier to attack however as the different key parts may be attacked independently which is not the case when they are obtained from the same intermediate key.

Resistance to lattice attacks is higher in embodiments that use a univariate private key polynomial that was obtained by a key material generation device that substituted

an identity number into multiple bivariate polynomials, reducing them modulo the reduction integer associated with said particular polynomial, and then summing the univariate polynomials. This is however not necessary.

In a simpler embodiment the key material generation device has a single bivariate polynomial, no private reduction integers only a public global reduction integer. Private key material is obtained by substituting the identity number into the bivariate polynomial and reducing modulo the public global reduction integer. The first network devices may use such a univariate polynomial with spacing to complicate lattice attacks.

Figure 1 is a schematic block diagram of a key material generation device 200 for configuring a network device for key sharing and a first network device 300;

Key generation device 200 is typically implemented as an integrated device. For example, key material generation device 200 may be comprised in a server. Key generation device 200 may configure network devices over a network, say a wireless network, or the internet, and the like. However, key material generation device 200 may also be integrated in a manufacturing device for manufacturing the network devices.

Key generation device 200 comprises a key material obtainer 210, a network device manager 230, and a polynomial manipulation unit 220. Key generation device 200 is intended to work with multiple network devices. Figure 1 shows one such device, first network device 300.

Key generation device 200 selects secret key material, also referred to as root key material. Key generation device 200 then derives local key material for each of the multiple network devices. The local key material is derived from the root key material and at least one public identity number A of the network device. In figure 1, network device 300 stores identity number 310. A network device may also store a further identity number and derive the identity number 310 therefrom when needed, e.g., by hashing the further identity number.

The local key material comprises parts that are private to a particular network device, i.e., only accessible to one particular network device and possibly trusted devices. The local key material may also contain parts that, though needed to obtain a shared key, are less critical to keep secret.

The use of the adjectives public and private, is intended as helpful for understanding: Even with access to all public data, the private data cannot be computed, at least not without unreasonable high resources given the security of the application or compared to the resources needed for key material generation, encryption, and decryption.

However, 'public' does not mean that the corresponding data is necessarily made available to anybody else than key material generation device 200 and the network devices. In particular, keeping the public global reduction integer and other public parameters secret from untrusted parties increases security. Likewise, access to private data may be restricted to the party that generated or needs that data, this increases security. However, a trusted party may be allowed access to the private data; Access to private data reduces security.

Using their local key material and the identity number of the other party, the network devices can agree on a shared key between them.

Key material obtainer 210 is configured to obtain in electronic form at least a first parameter set 250. Parameter set 250 comprises a public global reduction integer 256, N, a first private set of bivariate polynomials 252, $f_i(,)$, and a second private set of reduction integers 254, p_i, with each bivariate polynomial in the first set there is associated a reduction integer of the second set, and a public global reduction integer 256, N. The parameter set is generated for network nodes having identifying number of bit-size B. The parameter set will be used for generating local key material which in turn will be used to derive a shared key. The bit-size of the shared key b satisfies $b \leq B$. In an embodiment, $b < B$, in this way the amount of information that can be learned from the shared key is smaller than the amount of information that needs to be reconstructed. This makes the corresponding lattice problem harder.

The public global reduction integer of the parameter set 256, N is different from each of the reduction integers 254. Preferably, the public global reduction integer of parameter set 256, N is larger than each of the reduction integers 254 of that parameter set.

Key material obtainer 210 does not need interaction with a network device for obtaining the key material; in particular key material obtainer 210 does not need an identity number. Key generation device 200 may be a distributed system in which key material obtainer 210 is located at a different physical location than polynomial manipulation unit 220. Key material obtainer 210 generates all or part of the key material and/or obtains all or part of the key material from an external source. For example, key material obtainer 210 is suited to receive the public global reduction integer 256 from an external source and generate the first private set 252 and second set 254.

Key material obtainer 210 may comprise an electronic random number generator. The random number generator may be a true or pseudo random number generator. Key material obtainer 210 may generate a public global reduction integer, N, e.g., using the

electronic random number generator. Although, the public global reduction integer is public information, introducing randomness makes analyzing the system more difficult.

With each bivariate polynomial in a first set, a reduction integer from a second set is associated. The random coefficients may be randomly selected from the integers modulo the associated reduction integer.

Key material obtainer 210 may generate one or more coefficients of a reduction integer p_i in a second private set using the electronic random number generator. It is not necessary that the reduction integers are primes. However, they may be chosen as prime to increase resistance. Prime numbers give rise to fields, which is a species of rings. The same parameter set, i.e., the same first and second private sets, and public global reduction numbers, are used for all network devices that later need to share a key.

Key material obtainer 210 may generate one or more coefficients of a bivariate polynomial $f_i(,))$ in first private set 252, e.g., using the electronic random number generator. Key material obtainer 210 may generate all of the bivariate polynomial in this fashion. Key material obtainer 210 may use a maximum degree of these polynomials, say 2, or 3 or higher, and generate one more random coefficient than the degree. The maximum degree is higher for more secure applications, say, a maximum degree of 30, etc.

It is convenient to prescribe some aspects of first private sets 252 such as the number of polynomials in private sets 252 and the degrees of the polynomials, or the maximum degrees. It may also be prescribed that some of coefficients in the polynomials are zero, e.g., for reducing storage requirements.

First set 252 may contain two equal polynomials. This will work, however, unless the associated reduction integers are different the sets may be reduced in size. So typically, whenever two or more bivariate polynomials in the first set are the same, the associated reduction integers, i.e. the underlying rings, are different.

In an embodiment all first private sets of bivariate polynomials ($f_i(,)$) only comprise symmetric bivariate polynomials. Using only symmetric polynomials has the advantage that each network device can agree on a shared key with any other network device of the configured network devices. However, a first private set of bivariate polynomials may contain one or more asymmetric polynomials; this has the effect that the devices can be portioned into two groups: a device from one group can only agree on a shared key with a device of the second group.

Key material obtainer 210 is configured to obtain in electronic form a first private set of bivariate polynomials 252, also referred to as $f_i(,)$ in formulas. The

embodiment described below assumes that all bivariate polynomials in set 252 are symmetric.

A symmetric bivariate polynomial may also be notated as $f_i(x,y)$ with two formal variables as placeholder. A symmetric bivariate polynomial satisfies $f_i(x,y) = f_i(y,x)$. This requirement translates to a requirement on the coefficients, e.g., that the coefficient of a monomial $x^a y^b$ equals the coefficient of a monomial $x^b y^a$.

The number of polynomials in first private set 252 may be chosen differently depending on the application. The system will work when the first set contains only a single polynomial; in such a system keys may be successfully shared and provide a moderate level of security. However, the security advantage of mixing over different rings is only achieved when the first set has at least 2 polynomials in them, and the second set has at least two different reduction integers.

Private set 252 comprises at least one bivariate polynomial. In an embodiment of initiating key-agreement device 100 the private set 252 consists of one polynomial. Having only one polynomial in private set 252 reduces complexity, storage requirements and increases speed. However, having only one polynomial in private set 252 is considered less secure than having two or more polynomials in private set 252 because such a one-polynomial system does not profit from additional mixing in the summation described below. However, key sharing will work correctly and are considered sufficiently secure for low-value and/or low-security applications.

In the remainder, we will assume that private set 252 comprises at least two symmetric bivariate polynomials. In an embodiment, at least two, or even all of the polynomials are different; this complicates analysis of the system considerably. It is not necessary though, private set 252 may comprise two equal polynomials and still benefit from mixing in the summation step if these two polynomials are evaluated over different rings. Note that different reduction integers define different rings. In an embodiment, private set 252 comprises at least two equal polynomials associated with different associated reduction integers. Having two or more equal polynomials in the first set reduces storage requirements. In an embodiment, the second set comprises at least two polynomials, and all polynomials in the second set are different.

The polynomials in private set 252 may be of different degrees. With the degree of a symmetric bivariate polynomial we will mean the degree of the polynomial in one of the two variables. For example, the degree of $x^2y^2 + 2xy + 1$ equals 2 because the degree in x is 2. The polynomials may be chosen to have the same degree in each variable; if

the polynomials in private set 252 are symmetric the degree will be the same in the other variable.

The degrees of polynomials in private set 252 may be chosen differently depending on the application. Private set 252 comprises at least one symmetric bivariate polynomial of degree 1 or higher. In an embodiment, private set 252 comprises only polynomials of degree 1. Having only linear polynomials in private set 252 reduces complexity, storage requirements and increases speed. However, having only degree one polynomials in private set 252 is considered less secure than having at least one polynomial of degree at least two in private set 252 because such a system is considerably more linear. Even so, if multiple polynomials in private set 252 are evaluated over different rings, then the resulting encryption is not linear even if all polynomials in private set 252 are. In an embodiment, private set 252 comprises at least one, preferably two, polynomials of degree 2 or higher. However, key material generation, encryption, and decryption will work correctly if only degree 1 polynomials are used, and are considered sufficiently secure for low-value and/or low-security applications.

Having one or more polynomials in private set 252 with degree 0 will not impact the system, so long as the polynomial(s) with higher degree provide sufficient security.

For a mid-security application, private set 252 may comprise, or even consist of, two symmetric bivariate polynomials of degree 2. For a higher security application, private set 252 may comprise or even consist of two symmetric bivariate polynomials, one of degree 2 and one of degree higher than 2, say 3. Increasing the number of polynomials and/or their degrees will further increase security at the cost of increased resource consumption.

Preferably, the reduction integers are selected so that the difference of any two reduction integers in the same set of reduction integers has a common divisor. In particular, the common divisor may be 2^{b_1}; or in words, the difference between any two reduction integers ends in a least as many zero's as the size of the first bit-string.

For example, one way to generate the reduction integers and the public global reduction integer is as follows.

1. First generate the public global reduction integer N. For example as a random integer of prescribed size,
2. For each reduction integer p_i, generate integers $\beta_i^{(k)}$ with $1 \leq \beta_i^{(k)} \leq 2^B$ for $1 \leq k \leq t$.
3. Generate the reduction integer p_i as the difference

$$p_i = N - \sum_{k=1}^{t} \beta_i^{(k)} 2^{(\alpha+1)(k-1)B+\sum_{\ell=1}^{k} b_\ell}$$

The public global reduction integer may be chosen to have $t(\alpha + 1)B + b$ bits or more, wherein α is the highest degree in a single variable of the bivariate polynomials in the first private set.

Key material obtainer 210 may be programmed in software or in hardware or in a combination thereof. Key material obtainer 210 may share resources with polynomial manipulation unit 220 for polynomial manipulation.

Network device manager 230 is configured to obtain in electronic form an identity number 310, A for network device 300. Network device manager 230 may receive the identity number from the network device. For example, network device manager 230 may comprise or make use of a communication unit for receiving the identity number over a network. For example, network device manager 230 may comprise an antenna for receiving the identity number as a wireless signal. The identity number may be represented as a number of bits, typically, the number of bits in the identity number B is at least as large as the number of bits in the shared key.

Polynomial manipulation unit 220 is configured to compute a univariate private key polynomial 229 for a parameter set and an identifying number A. Polynomial manipulation unit 220 is applied to the parameter set of key material obtainer 210. The univariate private key polynomial that is thus obtained and the corresponding public global reduction integer are part of the local key material that will be sent to the network device.

Polynomial manipulation unit 220 receives the data in a parameter set from key material obtainer 210 over connection 238. Below it is described how polynomial manipulation unit 220 determines a univariate private key polynomial from the parameter set. Polynomial manipulation unit 220 may compute the univariate private key polynomial 229 as follows:

Univariate polynomials are obtained by substituting the identity integer A into each of the polynomials in the first private set of the parameter set that is currently processed. By substituting a value for only one variable of a bivariate polynomial, the bivariate polynomial reduces to a univariate polynomial. The resulting univariate polynomial is then reduced modulo the reduction integer associated with the bivariate polynomial in which the identity integer A was substituted. The resulting set of univariate polynomials is summed,

e.g., by adding the coefficients of equal powers of y in the polynomials. This may be obtained from the formula for C_i^A in: $KM^A(x) = \sum_{j=1}^{m} < f_j(x, A) >_{p_j} = \sum_i C_i^A x^i$

Suppose $f_i(x, y)$ is one of the bivariate polynomials in the first private set. The coefficients of this polynomial are taken from the ring $\mathbb{Z}_{p_i}$. That is the coefficients of the polynomials in the first set are taken from an integer ring. For simplicity, the variables x and y are used to represent the formal variables of the integers in the first set.

After substitution, polynomial manipulation unit 220 obtains $f_i(A, y)$. Polynomial manipulation unit 220 is further configured to reduce this term modulo p_i. Coefficients are reduced in the ring over which the system operates, e.g., Z_p, e.g., by reducing mod p. Preferably, polynomial manipulation unit 220 brings the result into a canonical form, i.e., a predetermined standardized representation. A suitable canonical form is representation of the coefficient sorted by degrees of the monomials. Alternatively, the substitution may be for y.

To ensure that the identity numbers act 'random' in the system a randomization step at a point in the chain is advisable to ensure that lattice attacks do not simplify. Especially if the network devices are given identity numbers according to a particular order, e.g., serial numbers, such a randomization step is advisable. For example, a cryptographic hash, say, sha-256 may be applied to the identity number, the result being shortened to B bits, before the substitution step.

Furthermore, identity numbers may be extended to more bits. For example, an identity number of B' bits may extended, e.g., by hashing and/or concatenation, to B bits, with $B' < B$. For example and identity number A may be extended to $H(A)$ or to $A||H(A)$; H denotes hashing and || denotes concatenation. The concatenation is done at the LSB side. A highly non-linear hash, such as a cryptographic hash is preferred for this operation.

If the first set only contains symmetric polynomials, then substitution of the identity integer A may be in either one of the two variables of the bivariate polynomial. However, if substitution is done in an asymmetric polynomial, more care is needed. For example polynomial manipulation unit 220 may be configured to obtain whether first network device 300 is in a first or second group. The first and second groups are associated with the first and second variable of the bivariate polynomials, respectively. For a network device in the first group always the first variable is used. For a network device in the second group always the second variable is used.

Figure 1 shows one possible way to implement this function. Figure 1 shows a substituting unit 222, a polynomial reduction unit 224, a polynomial addition unit 226 and a

sum of a set of univariate polynomials 228; the latter will be univariate private key polynomial 229. These may work as follows. Substituting unit 222 substitutes the identity integer A into a bivariate polynomial of the first set. Substituting unit 222 may collect terms to bring the result in canonical form, but this may also wait. Polynomial reduction unit 224 receives the result of the substitution and reduces it modulo the reduction integer associated with the bivariate polynomial in which was substituted.

The result of substituting the identity integer A into said particular polynomial $f_i(A, y)$ and reducing modulo the reduction integer associated with said particular polynomial is represented as a list of coefficients in a canonical form before the summing by polynomial addition unit 226. The variable y acts as a formal variable. This substitution is sometime notated simply as: $f_i(A,)$.

Polynomial addition unit 226 receives the reduced univariate polynomials and adds them to a running total in sum 228. Sum 228 was reset to 0 prior to the generation of the univariate private key polynomial. Polynomial addition unit 226 may add the polynomials coefficient-wise, using either natural arithmetic or modulo the public global reduction number associated to the parameter set.

When all polynomials of the first private set are processed in this way, the result in sum 228 may be used as the univariate private key polynomial. The resulting univariate private key polynomial, say in sum 228, may be represented as a list of coefficients and in a canonical form.

Network device manager 230 is further configured for electronically storing the generated univariate private key polynomial 229 and the corresponding public global reduction integer 256, N at the network device. Using the univariate private key polynomial 229 and its identity number or numbers, first network device 300 can share keys with other devices configured from the same root material. Network device manager 230 may also be configured for electronically storing the parameters B and b at the network device.

Although polynomial manipulation unit 220 may be implemented in software, polynomial manipulation unit 220 is particularly suited for implementation in hardware. If only polynomial reduction unit 224 is implementing hardware a significant speed improvement will be obtained; part of the functionality of key material generation device 200 that is not performed by a hardware version of the unit 224 may be performed in software running of a processor.

Figure 1 shows polynomial manipulation unit 220 receiving an identity number message 232 from first network device 300; first network device 300 receiving a

public global reduction integer message 234 from key material obtainer 210 and a univariate private key polynomial message 236 from polynomial manipulation unit 220. These messages typically are sent and received through network device manager 230. Univariate private key polynomial message 236 and public global reduction integer message 234 may be combined in a single message. The public global reduction integer message 234 contains the public global reduction integer, corresponding to the univariate private key polynomial in the univariate private key polynomial message 236. Identity number message 232 may contain the identity number. Identity number message 232 may also or instead contain a further identity number, key material generation device 200 being configured to derive the identity number from the one or more further identity numbers, e.g., by hashing them.

Key generation device 200 may be configured to obtain an identity number by generating an identity number for first network device 300. Such a configuration is well suited to a manufacturing facility. In that case first network device 300 receives identity number message 232 from configuration key material generation device 200, instead of sending it, say receive identity number message 232 from key material obtainer 210 or polynomial manipulation unit 220.

Figure 2a is a schematic block diagram of a first network device 300 and a second network device 350. First network device 300 and second network device 350 are configured to determine a shared key together.

Second network device 350 may be of the same design as network device 300. We only describe first network device 300 in detail, second network device 350 may be the same or similar. Figure 2a only shows that second network device 350 stores an identity number 355. The identity number 355 of second network device 350 is public and may be exchanged with network device 300 to share a key. Second network device 350 also needs local key material (not shown), in particular a univariate private key polynomial corresponding to identity number 355.

First network device 300 comprises an electronic storage 320, a communication unit 342, a polynomial manipulation unit 330 and a key derivation device 340.

Storage 320 stores local key material of device 300.; local key material comprises a univariate polynomial univariate private key polynomial and a public global reduction integer. In the embodiment shown in figure 2a, the device 300 comprises a set of key material 370. Key material 370 comprises univariate private key polynomial 372 and a public global reduction integer 374.

Storage 320 also stores the identity number 310, A, that was used to generate the univariate private key polynomial in the key material.

Storage 320 may be a memory, say a non-volatile and writable memory, such as flash memory. Storage 320 may be other types of storage, say magnetic storage such as a hard disk. Storage 320 may be write-once memory.

Communication unit 342 is configured to obtain the identity number 355 of second network device 350. Communication unit 342 may be implemented as a wired connection, say a Wi-Fi, Bluetooth or ZigBee connection. Communication unit 342 may be implemented with a connection over a data network, say the internet.

Polynomial manipulation unit 330 is configured to derive a shared key with device 350 using the key material in storage 320. Device 350 also has key materials corresponding to the same root key material as device 300. Device 300 may receive the identity number B from device 350. Device 300 may also receive a further identity number and derive the identity number therefrom. Below it is described how polynomial manipulation unit 330 may derive a shared key using first key material 370.

Polynomial manipulation unit 330 may comprise a substituting unit 332, and an integer reduction unit 334.

Polynomial manipulation unit 330 is configured to substitute the identity integer B, say of device 350, into the univariate private key polynomial 372 and reduce the result of the substitution modulo the public global reduction integer 374; thus obtaining an intermediate key. Polynomial manipulation unit 330 may use similar hardware or software as substituting unit 222 and polynomial reduction unit 224. Note that first network device 300 does not have access to the first and second private set 252, 254.

In an embodiment, key derivation unit 340 is configured to extract from the intermediate key multiple bit-strings, the multiple bit-strings being non-overlapping, consecutive substrings of the intermediate key, each two adjacent bit-strings of the multiple bit-strings are separated in the intermediate key by at least one bit which does not belong to any of the multiple bit-strings, and derive the shared key from the multiple bit-strings.

For example, consider three bit-strings ($t = 3$). The binary representation of the intermediate key K may comprise $a = 3s + b_1 + b_2 + b_3$ bits (assuming constant spacing s). Let us say that K has binary representation $k_{a-1}k_{a-2} \dots k_0$. Where k_0 is the least significant bit. The bits in the first bit-string are the bits $k_{b_1-1} \dots k_0$. The bits in the second bit string are the bits $k_{b_1+b_2+s-1} \dots k_{b_1+s}$. The bits in the third bit string are the bits $k_{b_1+b_2+b_3+2s-1} \dots k_{b_1+b_2+2s}$. The

top s bits of the intermediate key are also spacing and not used anyone of the multiple bit-strings.

The spacing is not necessarily constant, although this appears to give better reconciliation. The multi-bit strings each have at least length 1. To obtain best distribution over the different bit-strings, they may have equal lengths.

The key derivation unit may directly derive a key from the multiple-bit strings and accept that there is a chance that the shared keys derived at device A and B are not the same. For some applications this is acceptable.

Optionally network device 300 comprises a key-reconciliation unit 336; shown in figure 2a as part of the key derivation unit 340. It may happen that device 300 and device 350 do not arrive at the same shared key. An application may chose to ignore this possibility. In doing so, some pairs of network devices may not be able to engage in encrypted and/or authenticated communication as they lack a common shared key. For some applications it is sufficient that only some pairs of network devices are secured, e.g., ad-hoc networks are an example of this. Devices 300 and 350 may also be configured with an optional key-reconciliation unit 336. In one of the two devices 300 and 350 the key-reconciliation unit 336 generates key-reconciliation data from the generated key and sends it to the other device; in the other device key-reconciliation unit 336 uses received key-reconciliation data to adapt the generated shared key so that the shared key derived in both devices is the same.

If key-reconciliation unit 336 is used to adapt keys, it adapts the generated shared key until it conforms to the key-reconciliation data, i.e., deriving key-reconciliation data from the adapted shared key would give the same result as the received key-reconciliation data for that key. If the least significant bits are used as confirmation data, the key-reconciliation unit adds multiples until the c least significant bits are the same as the received bits.

Instead of sending and receiving key-reconciliation data per bit-string key, the key-reconciliation unit may also be configured to generate key-reconciliation data over the assembled large shared key, possibly even after a key confirmation algorithm like KDF. In this case, the key-reconciliation unit adapts all bit-strings keys simultaneously until a large key is found that satisfies the key-reconciliation data. Although varying multiple bit-strings keys at the same is more work, generating key-reconciliation data over the large key is also much more secure as less direct information is available for the bit-strings keys.

After reconciliation key derivation device 340 may use, e.g., the concatenation of the multiple bit-strings directly as a key. Key derivation device 340 may also apply a key

derivation function to the shared key, for example the function KDF, defined in the OMA DRM Specification of the Open Mobile Alliance (OMA-TS-DRM-DRM-V2_0_2-20080723-A, section 7.1.2 KDF) and similar functions.

Figure 2a further shows an optional cryptographic unit 345 in first network device 300. Cryptographic unit 345 is configured to use the shared key. For example, cryptographic unit 345 may be an encryption unit configured for encrypting an electronic message with the shared symmetric key. For example, cryptographic unit 345 may be a decryption unit configured for decryption an electronic message with the shared symmetric key.

Typically, the device 200 and the 300 each comprise a microprocessor (not shown) which executes appropriate software stored at the device 200 and the 350; for example, that software may have been downloaded and/or stored in a corresponding memory, e.g., a volatile memory such as RAM or a non-volatile memory such as Flash (not shown). Alternatively, the devices 200 and 300 may, in whole or in part, be implemented in programmable logic, e.g., as field-programmable gate array (FPGA). Devices 200 and 300 may be implemented, in whole or in part, as a so-called application-specific integrated circuit (ASIC), i.e. an integrated circuit (IC) customized for their particular use. For example, the circuits may be implemented in CMOS, e.g., using a hardware description language such as Verilog, VHDL etc.

In an embodiment, first network device 300 comprises a storage circuit, communication circuit, a polynomial manipulation circuit, and a key derivation circuit. The device 300 may comprise additional circuits, e.g., a key-reconciliation circuit. In an embodiment, key material generation device 200 comprises a key material obtainer circuit, a network device manager circuit, a polynomial manipulation circuit. The circuits implement the corresponding units described herein. The circuits may be a processor circuit and storage circuit, the processor circuit executing instructions represented electronically in the storage circuits. The circuits may also be, FPGA, ASIC or the like.

Figure 3a is a schematic block diagram of a key sharing system 100.

Key sharing system 100 comprises key material generation device 200, and multiple network devices; shown are network device 300, 350 and 360. The network devices each receive an identity number, univariate private key polynomial and the global reduction integer from key material generation device 200. Using this information they can agree on a shared key. For example, first network device 300 and second network device 350 each send their identity number to the other party. They can then compute a shared key. Someone with

knowledge of the communication between first network device 300 and second network device 350 and even the global reduction integer cannot obtain their shared key, without using unreasonable large resources. Not even device 360 can derive the key shared between devices 300 and 350.

Figure 3b is a schematic block diagram of a similar key sharing system 102. System 102 is the same as system 100 except that the network devices receive their identity number from a configuration server 110, also referred to as a personalization device. The network devices then register with key material generation device 200 by sending their identity number. The configuration server 110 may assign an identity number that is also used for other purposes. For example, configuration server 110 may assign a network address, such as a MAC address. The network address is used by the network node for routing network traffic from a second network node to itself. However, the network address may also be used as the identity number. In this case, the network node makes its network address available to key material generation device 200 and receives a univariate private key polynomial which allows the network node to engage in encrypted communication using its network address as identity number. It is preferred that an identity number has full entropy, i.e., B bits of entropy. However, when this cannot be realized, it is preferred to perform an entropy smoothing function, e.g., a hash function before using the number as the identity number.

The configuration server 110 may generate identity numbers to increase security of the system by avoiding identity numbers that are close, i.e., that share many or all of the most significant bits. For example, server 110 may generate the identity numbers randomly, say true or pseudo random. It is also sufficient to append predetermined number of random bits to an identity number, say 10 bits. The identity number may have the form $A_1 || A_2$, in which A_1 is not random, say a serial number, network address, or the like, and wherein A_2 is random. A_2 may be generated by a random number generator. A_2 may also be generated by hasing A_1. If a keyed hash is used, say an HMAC, this then A_2 is indistinguishable from random to parties without access to said key. The key may be generated and stored by server 110.

Server 110 may be included in key material generation device 200, e.g., incorporated in network manager 230.

Figure 4a is schematic block diagram of an integrated circuit 400 which may be configured as a key material generation device or network device. Integrated circuit 400 comprises a processor 420, a memory 430, and an I/O unit 440. These units of integrated

circuit 400 can communicate amongst each other through an interconnect 410, such as a bus. Processor 420 is configured to execute software stored in memory 430 to execute a method as described herein. In this way integrated circuit 400 may be configured as key material generation device 200 or as a network device, such as first network device 300; Part of memory 430 may store a public global reduction integer, first private sets of bivariate polynomials, second private sets of reduction integers, an identity number, a plain message and/or encrypted message as required.

I/O unit 440 may be used to communicate with other devices such as devices 200, or 300, for example to receive key data, such as first private set of bivariate polynomials 252 and possibly associated parameters, such as sizes, degrees, moduli and the like, or to send and receive encrypted and/or authenticated messages. I/O unit 440 may comprise an antenna for wireless communication. I/O unit 440 may comprise an electric interface for wired communication.

Integrated circuit 400 may be integrated in a computer, mobile communication device, such as a mobile phone, etc. Integrated circuit 400 may also be integrated in lighting device, e.g., arranged with an LED device. For example, an integrated circuit 400 configured as a network device and arranged with lighting unit such as an LED, may receive commands encrypted with a shared symmetric key.

Multiple network devices, say incorporated in a lighting device, may form the nodes of an encrypted network, in which links are encrypted using shared keys between the nodes.

Although polynomial manipulation may be performed by processor 420 as instructed by polynomial manipulation software stored in memory 430, the tasks of key material generation, and calculating the univariate polynomials are faster if integrated circuit 400 is configured with optional polynomial manipulation unit 450. In this embodiment, polynomial manipulation unit 450 is a hardware unit for executing substitution and reduction operations.

Figure 4b shows a computer readable medium 1000 having a writable part 1010 comprising a computer program 1020, the computer program 1020 comprising instructions for causing a processor system to perform a method of key sharing or key material generation, according to an embodiment. The computer program 1020 may be embodied on the computer readable medium 1000 as physical marks or by means of magnetization of the computer readable medium 1000. However, any other suitable embodiment is conceivable as well. Furthermore, it will be appreciated that, although the

computer readable medium 1000 is shown here as an optical disc, the computer readable medium 1000 may be any suitable computer readable medium, such as a hard disk, solid state memory, flash memory, etc., and may be non-recordable or recordable. The computer program 1020 comprises instructions for causing a processor system to perform said method of key sharing or key material generation.

Figure 5 schematically shows a flowchart illustrating a key sharing method 500. Method 500 may be executed by first network device 300 to share a key with second network device 350. Key sharing method 500 is arranged for a first network device to determine a shared key with a second network device 350

Method 500 comprises:

- storing 502 a univariate private key polynomial 372 and a public global reduction integer 374, N obtained from an external key material generation device arranged to configure at least the first and second network device for key sharing, and storing a first identity number 310, A used by the key material generation device to generate the univariate private key polynomial 372,
- obtaining 504 a second identity number 355 of the second network device,
- substituting 506 the second identity number into the univariate private key polynomial,
- reducing 508 the result of the substituting modulo the public global reduction integer N thus obtaining an intermediate key, and
- extracting 510 from the intermediate key multiple bit-strings, the multiple bit-strings being non-overlapping, consecutive substrings of the intermediate key, each two adjacent bit-strings of the multiple bit-strings are separated in the intermediate key by at least one bit which does not belong to any of the multiple bit-strings, and
- deriving 512 the shared key from the multiple bit-strings.

Figure 6 schematically shows a flowchart illustrating a key material generation method. Method 600 may be executed by key material generation device 200. Key generation method 600 configures a first network device 300 for sharing a shared key, the shared key being b bits long,

the key material generation method comprising:

- obtaining 602 in electronic form a first private set of bivariate polynomials 252, $f_i(,)$, and a second private set of reduction integers 254, p_i, with each bivariate polynomial in the first set there is associated a reduction integer of the second set, and a public global reduction integer 256, N, the public global reduction integer has at least

$t(\alpha + 1)B + b$ bits, wherein α is the highest degree in a single variable of the bivariate polynomials in the first private set, and t is the number of bit-strings in the multiple bit-strings,

- obtaining 604 in electronic form an identity number 310, A for the network device, the identity number being B bits long, and
- computing 606 for the network device a univariate private key polynomial 229 from the first and second private sets by
  - obtaining 608 a set of univariate polynomials by
    - for each particular polynomial of the first private set, substituting the identity number A into said particular polynomial f_i, and reducing modulo the reduction integer associated with said particular polynomial, and
  - summing the set of univariate polynomials,
- storing 610 the generated univariate private key polynomial 229, 236 and the public global reduction integer 256, N at the first network device.

Many different ways of executing the method are possible, as will be apparent to a person skilled in the art. For example, the order of the steps can be varied or some steps may be executed in parallel. Moreover, in between steps other method steps may be inserted. The inserted steps may represent refinements of the method such as described herein, or may be unrelated to the method. For example, steps 608 may be executed, at least partially, in parallel. Moreover, a given step may not have finished completely before a next step is started.

A method according to the invention may be executed using software, which comprises instructions for causing a processor system to perform method 500 and 600. Software may only include those steps taken by a particular sub-entity of the system. The software may be stored in a suitable storage medium, such as a hard disk, a floppy, a memory etc. The software may be sent as a signal along a wire, or wireless, or using a data network, e.g., the Internet. The software may be made available for download and/or for remote usage on a server. A method according to the invention may be executed using a bitstream arranged to configure programmable logic, e.g., a field-programmable gate array (FPGA), to perform the method.

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. An embodiment relating to a computer program product comprises computer executable instructions corresponding to each of the processing steps of at least one of the methods set forth. These instructions may be subdivided into subroutines and/or be stored in one or more files that may be linked statically or dynamically. Another embodiment relating to a computer program product comprises computer executable instructions corresponding to each of the means of at least one of the systems and/or products set forth.

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.

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.

In the claims references in parentheses refer to reference signs in drawings of embodiments or to formulas of embodiments, thus increasing the intelligibility of the claim. These references shall not be construed as limiting the claim.

List of Reference Numerals in figures 1, 2a, 3a, 3b

100,102	a key sharing system
110	a personalization device
200	a key material generation device
210	a key material obtainer
220	a polynomial manipulation unit
222	a substituting unit
224	a polynomial reduction unit
226	a polynomial addition unit
228	sum of a set of univariate polynomials
229	univariate private key polynomial
230	a network device manager
232	an identity number message
234	a public global reduction integer message
236	a univariate private key polynomial message
250	a parameter set
252	a first private set of bivariate polynomials
254	a second private set of reduction integers
256	a public global reduction integer
300	a first network device
310	an identity number
320	an electronic storage
330	a polynomial manipulation unit
332	a substituting unit
334	an integer reduction unit
336	a key-reconciliation unit
340	a key derivation device
342	a communication unit
345	a cryptographic unit
350	a second network device
355	an identity number
360	a third network device
370	a key material

372 a univariate private key polynomial

374 a public global reduction integer

CLAIMS:

1. A first network device (300) arranged to determine a shared key with a second network device (350), the first network device comprising

- an electronic storage (320) storing a univariate private key polynomial (372) and a public global reduction integer (374, N) obtained from an external key material generation device arranged to configure at least the first and second network device for key sharing, the storage further storing a first identity number (310, A) for the first network device used by the key material generation device to generate the univariate private key polynomial (372),

- a communication unit (342) arranged to obtain a second identity number (355) of the second network device, the second network device being different from the first network device,

- a polynomial manipulation unit (330) arranged to

  - substitute the second identity number into the univariate private key polynomial,

  - reduce the result of the substituting modulo the public global reduction integer (N) thus obtaining an intermediate key, and

- a key derivation unit arranged to

  - extract from the intermediate key multiple bit-strings, the multiple bit-strings being non-overlapping, consecutive substrings of the intermediate key, each two adjacent bit-strings of the multiple bit-strings are separated in the intermediate key by at least one bit which does not belong to any of the multiple bit-strings, and

  - derive the shared key from the multiple bit-strings.

2. A first network device as in Claim 1, wherein

- each two adjacent bit-strings of the multiple bit-strings are separated in the intermediate key by a same spacing number of bits (s).

3. A first network device as in any one of the preceding claims, wherein all of the multiple bit-strings have the same size.

4. A first network device as in any one of the preceding claims, wherein a first bit string of the multiple bit strings includes the least significant bit of the intermediate key.

5. A first network device as in any one of the preceding claims, wherein a last bit string of the multiple bit strings does not include the most significant bit of the intermediate key.

6. A first device (300) as in any one of the preceding claims, wherein

- the communication unit (342) is further arranged to receive key-reconciliation data from the second device, the first device comprising a key-reconciliation unit (336) arranged to modify the multiple bit-strings to conform to the received key-reconciliation data, the shared key being derived from the modified multiple bit-strings,

or

- the first device comprising a key-reconciliation unit (336) arranged to compute key-reconciliation data from the multiple bit-strings, the communication unit (342) being further arranged to send the key-reconciliation data to the second device.

7. A first device (300) as in any one of the preceding claims, wherein the univariate private key polynomial (372) has previously been obtained by:

- obtain in electronic form a bivariate polynomial, and a public global reduction integer (256, N),
- obtain in electronic form an identity number (310, A) for the first network device,
- compute for the first network device a univariate private key polynomial (229) by substituting the identity number (A) into the bivariate polynomial and reducing modulo the public global reduction integer.

8. A first device (300) as in any one of the preceding claims, wherein the univariate private key polynomial (372) has previously been obtained by:

- obtain in electronic form a first private set of bivariate polynomials (252, $f_i(,)$), and a second private set of reduction integers (254, p_i), with each bivariate polynomial in the first set there is associated a reduction integer of the second set, and a public global reduction integer (256, N),

- obtain in electronic form an identity number (310, A) for the network device, and
- compute for the network device a univariate private key polynomial (229) from the first and second private sets by
  - obtaining a set of univariate polynomials by
    - for each particular polynomial of the first private set, substituting the identity number (A) into said particular polynomial $f_i(A,)$ and reducing modulo the reduction integer associated with said particular polynomial, and
  - summing the set of univariate polynomials and reducing modulo the public global reduction integer.

9. A first network device as in Claim 8, wherein the spacing number of bits (s) has $(\alpha + 1)B$ bits or more, wherein
   - α is the highest degree in a single variable of the bivariate polynomials in the first private set
   - the first and second identity numbers have an identity number length B bits.

10. A first device (300) as in Claim 8 or 9, wherein the public global reduction integer has $ts + b$ bits or more, wherein
   - t is the number of bit-strings in the multiple bit-strings, t being larger than one
   - s is the spacing number of bits
   - the multiple bit-strings together have a key length b bits.

11. A first device as in any one of Claims 8, 9 and 10, wherein each private reduction integer

$$p_i = N - \sum_{k=1}^{t} \beta_i^{(k)} 2^{(\alpha+1)(k-1)B+\sum_{\ell=1}^{k} b_\ell}$$

for some integers $\beta_i^{(k)}$ with $1 \leq \beta_i^{(k)} \leq 2^B$ for $1 \leq k \leq t$.

12. A key sharing system comprising a first device (300) as in any one of the preceding claims and a key material generation device (200), wherein the key material generation device (200) is arranged to configure a first network device (300) for sharing a

shared key, the key generation device comprising:

- a key material obtainer (210) arranged to
  - obtain in electronic form a bivariate polynomial, and a public global reduction integer (256, N),
- a network device manager (230) arranged to obtain in electronic form an identity number (310, A) for the first network device,
- a polynomial manipulation unit (220) arranged to compute for the network device a univariate private key polynomial (229) by
  - computing for the first network device a univariate private key polynomial (229) by substituting the identity number (A) into the bivariate polynomial and reducing modulo the public global reduction integer
- the network device manager being further arranged to electronically store the generated univariate private key polynomial (229, 236) and the public global reduction integer (256, N) at the first network device.

13. A key sharing system as in Claim 12, wherein the shared key is b bits long, and the identity number is B bits long

- the key material obtainer (210) being arranged to
  - obtain in electronic form a first private set of bivariate polynomials (252, $f_i(,)$), and a second private set of reduction integers (254, p_i), with each bivariate polynomial in the first set there is associated a reduction integer of the second set, and a public global reduction integer (256, N), the public global reduction integer has at least $t(\alpha+1)B+b$ bits, wherein α is the highest degree in a single variable of the bivariate polynomials in the first private set, and t is the number of bit-strings in the multiple bit-strings,
- the polynomial manipulation unit (220) being arranged to compute for the network device a univariate private key polynomial (229) from the first and second private sets by
  - obtaining a set of univariate polynomials by
    - for each particular polynomial of the first private set, substituting the identity number (A) into said particular polynomial $f_i(A,)$ and reducing modulo the reduction integer associated with said particular polynomial, and
  - summing the set of univariate polynomials.

14. A key sharing method (500) arranged for a first network device to determine a shared key with a second network device (350), the method comprising

- storing (502) a univariate private key polynomial (372) and a public global reduction integer (374, N) obtained from an external key material generation device arranged to configure at least the first and second network device for key sharing, and storing a first identity number (310, A) used by the key material generation device to generate the univariate private key polynomial (372),
- obtaining (504) a second identity number (355) of the second network device,
- substituting (506) the second identity number into the univariate private key polynomial,
- reducing (508) the result of the substituting modulo the public global reduction integer (N) thus obtaining an intermediate key, and
- extracting (510) from the intermediate key multiple bit-strings, the multiple bit-strings being non-overlapping, consecutive substrings of the intermediate key, each two adjacent bit-strings of the multiple bit-strings are separated in the intermediate key by at least one bit which does not belong to any of the multiple bit-strings, and
- deriving (512) the shared key from the multiple bit-strings.

15. A key sharing system method (600) comprising a key material generation method to configure a first network device (300) for sharing a shared key and the key sharing method of Claim 14, the shared key being b bits long, the key material generation method comprising:

- obtaining (602) in electronic form a first private set of bivariate polynomials (252, $f_i(,)$), and a second private set of reduction integers (254, p_i), with each bivariate polynomial in the first set there is associated a reduction integer of the second set, and a public global reduction integer (256, N), the public global reduction integer has at least $t(\alpha+1)B+b$ bits, wherein α is the highest degree in a single variable of the bivariate polynomials in the first private set, and t is the number of bit-strings in the multiple bit-strings,
- obtaining (604) in electronic form an identity number (310, A) for the network device, the identity number being B bits long, and
- computing (606) for the network device a univariate private key polynomial (229) from the first and second private sets by
  - obtaining (608) a set of univariate polynomials by

- for each particular polynomial of the first private set, substituting the identity number (A) into said particular polynomial $f_i(A,)$ and reducing modulo the reduction integer associated with said particular polynomial, and
- summing the set of univariate polynomials,
- storing (610) the generated univariate private key polynomial (229, 236) and the public global reduction integer (256, N) at the first network device.

16. A computer program (1020) comprising computer program instructions arranged to perform the method of claim 14 or 15 when the computer program is run on one or more computers.

17. A computer readable medium (1000) comprising the computer program (1020) as in claim 16.

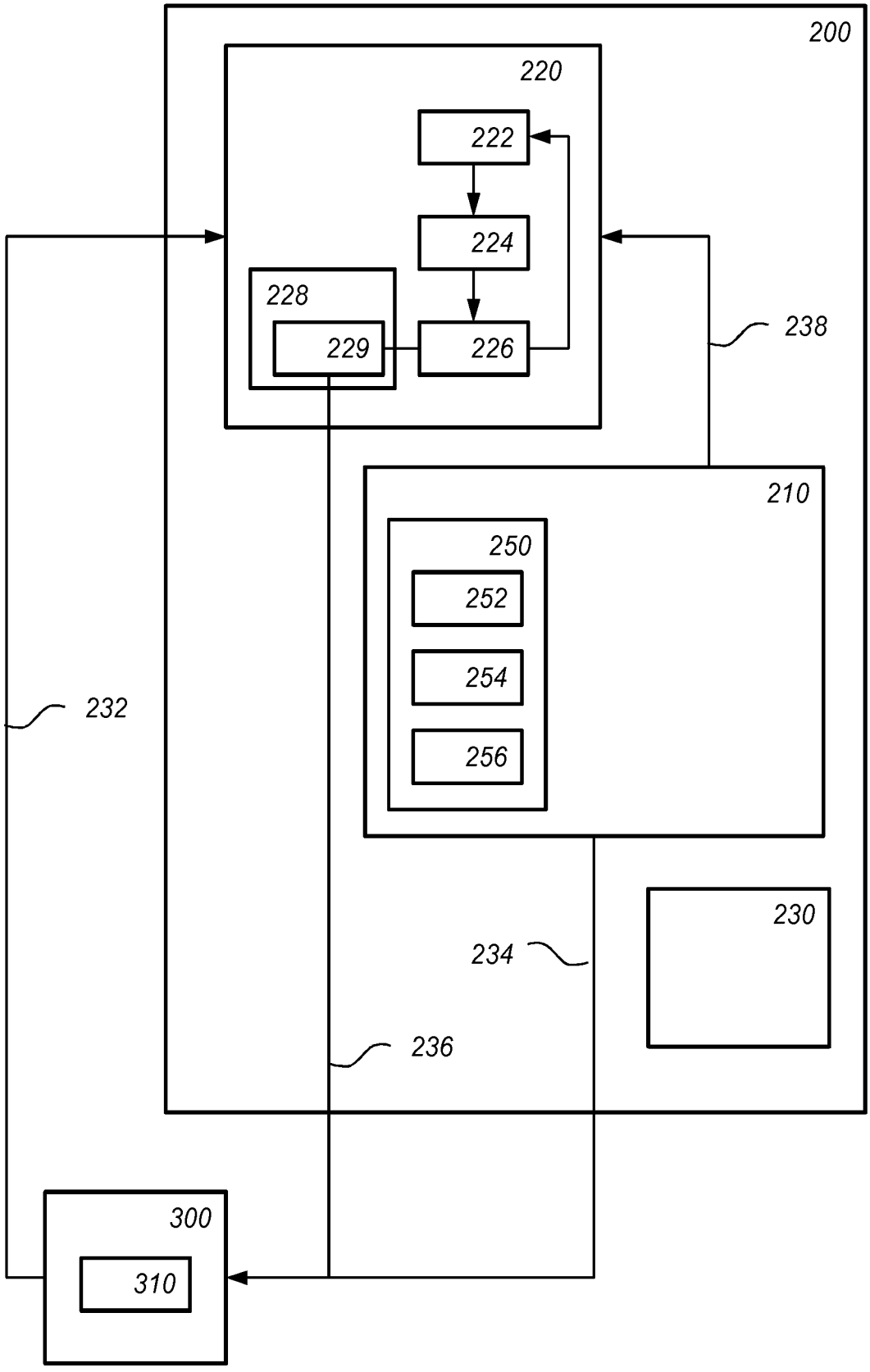
200
220
222
224
228
229
226
238
210
250
252
254
256
232
230
234
236
300
310

Fig. 1

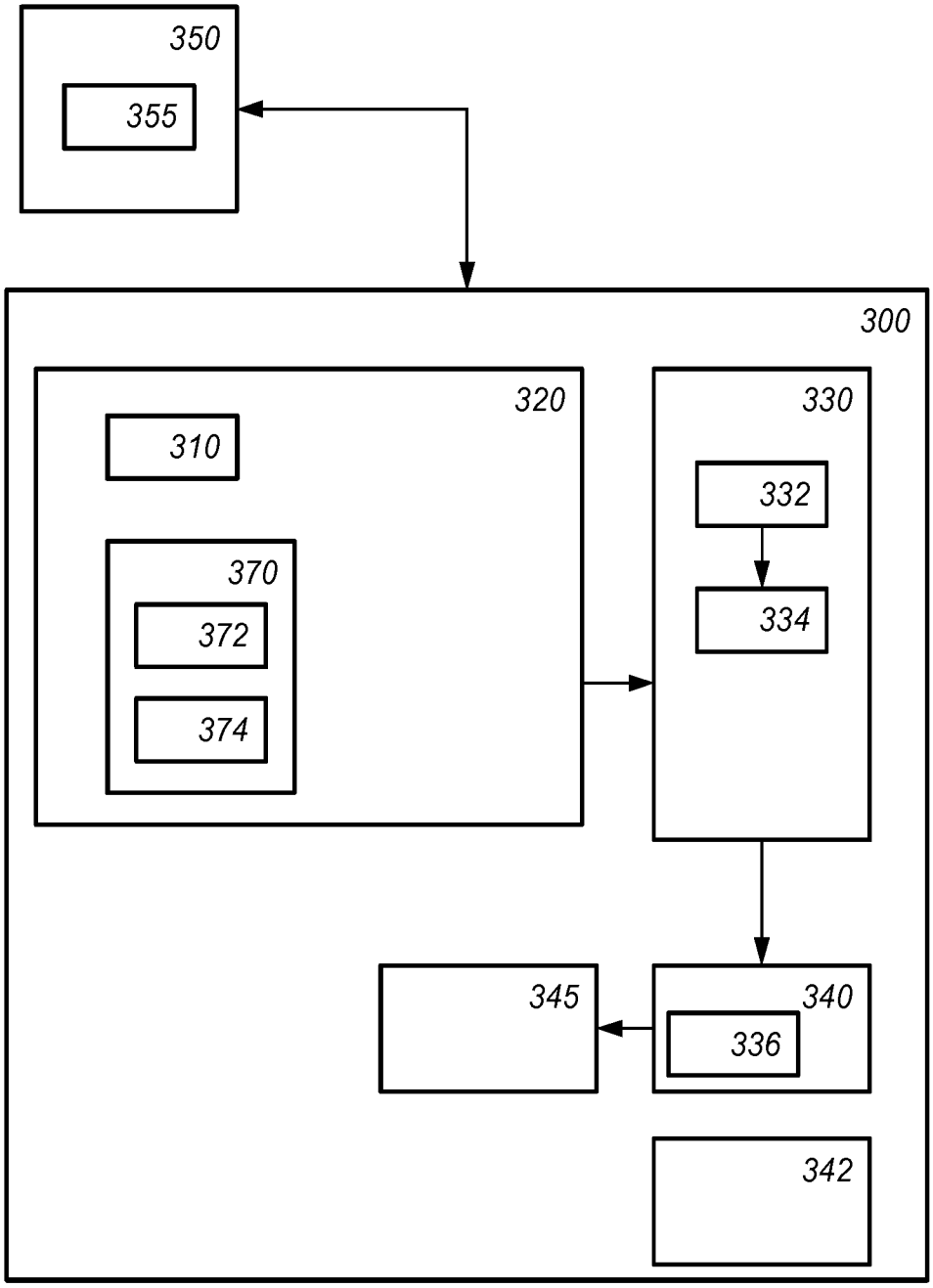

Fig. 2a

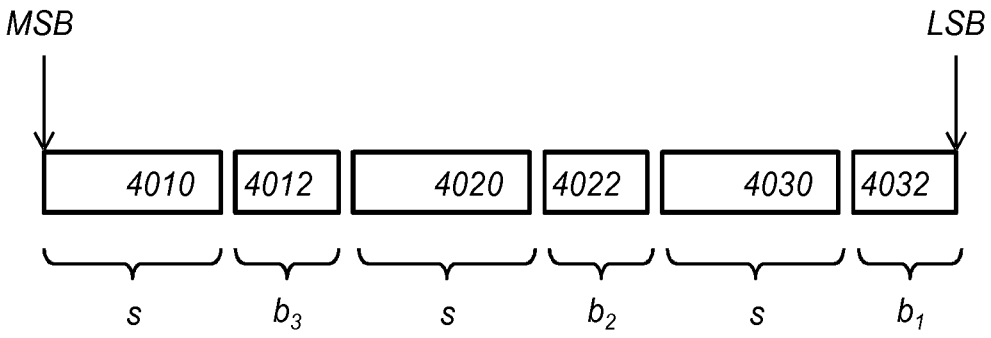

Fig. 2b

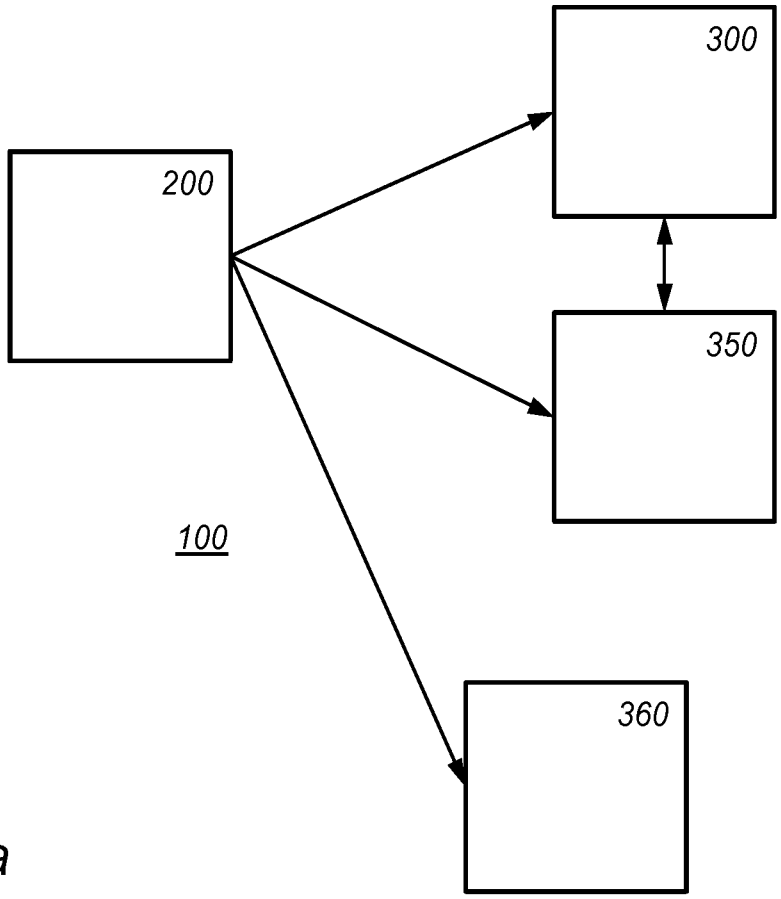

Fig. 3a

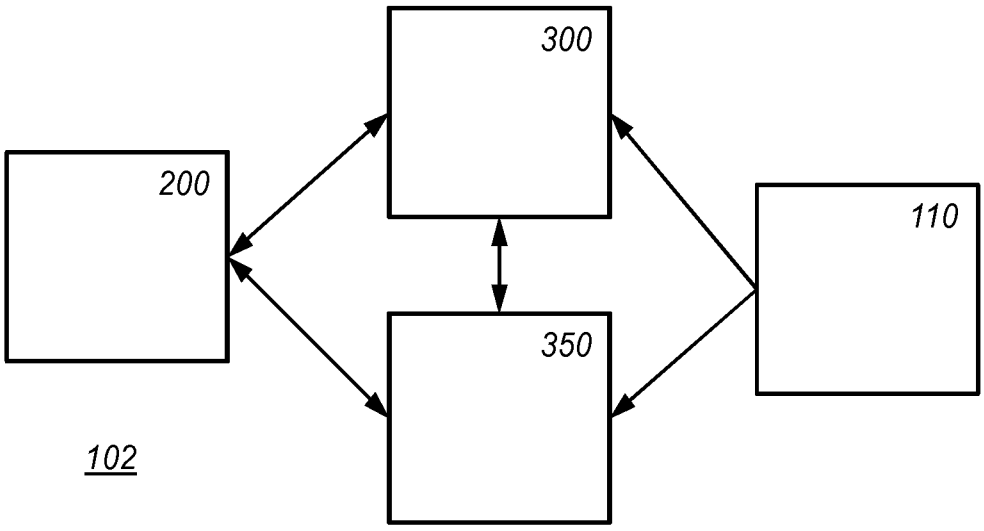

Fig. 3b

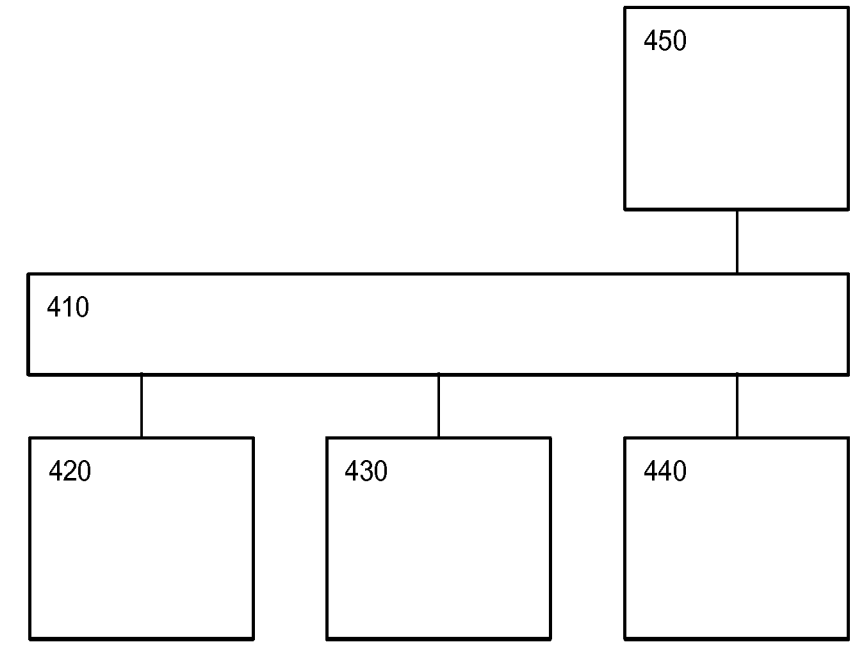

Fig. 4a

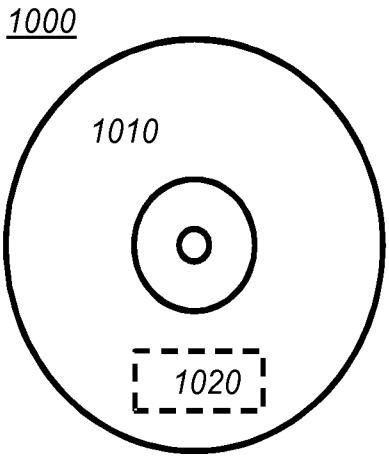

Fig. 4b

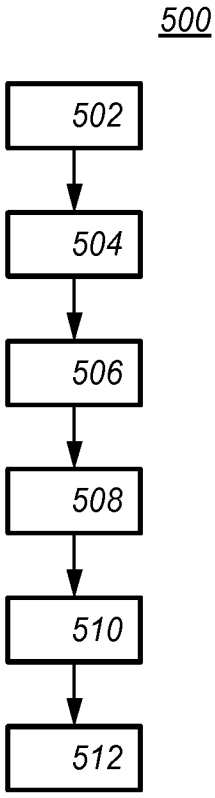

Fig. 5

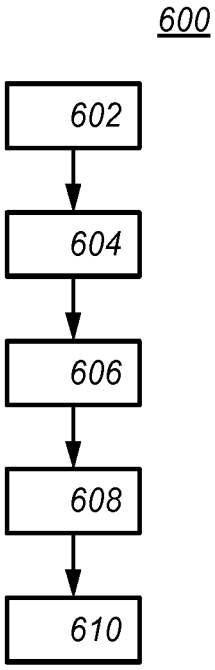

Fig. 6

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2016/069133

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04L9/08
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
EPO-Internal, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	WO 2013/174554 A1 (KONINKL PHILIPS NV [NL]) 28 November 2013 (2013-11-28) cited in the application pages 17-23	1-17
A	OSCAR GARCIA MORCHON ET AL: "Towards fully collusion-resistant ID-based establishment of pairwise keys", INTERNATIONAL ASSOCIATION FOR CRYPTOLOGIC RESEARCH,, vol. 20121128:171246, 28 November 2012 (2012-11-28), pages 1-13, XP061006992, the whole document	1-17

-/--

[X] Further documents are listed in the continuation of Box C.

[X] See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
9 November 2016	21/11/2016

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Horbach, Christian

1

Form PCT/ISA/210 (second sheet) (April 2005)

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2016/069133

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	WO 2015/004065 A1 (KONINKL PHILIPS NV [NL]) 15 January 2015 (2015-01-15) page 3, lines 18-31 page 5, lines 15-19 -----	1-17
A	US 2009/060188 A1 (MCGREW DAVID [US] ET AL) 5 March 2009 (2009-03-05) figures 2,4 -----	1-17

1

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/EP2016/069133

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 2013174554 A1	28-11-2013	CN 104303451 A	21-01-2015
		EP 2667539 A1	27-11-2013
		EP 2853057 A1	01-04-2015
		ES 2556027 T3	12-01-2016
		JP 5755391 B2	29-07-2015
		JP 2015521001 A	23-07-2015
		US 2016254909 A1	01-09-2016
		WO 2013174554 A1	28-11-2013
WO 2015004065 A1	15-01-2015	CN 105359455 A	24-02-2016
		EP 3020159 A1	18-05-2016
		JP 2016524431 A	12-08-2016
		US 2016149708 A1	26-05-2016
		WO 2015004065 A1	15-01-2015
US 2009060188 A1	05-03-2009	NONE	