Chapter 5
Public Key Cryptology - Part II

February 15, 2010

5

The basic concepts of PKC become widely known in the later 1970's and many schemes involving its use have been developed since then. Most of these were based on modular arithmetic (see Chapter 3) but some used other techniques such as Lucas series and Elliptic Curves (see Chapter 10).

The central requirements for public key encryption and digital signatures could both be met by RSA, but at the time there were three problems with that approach.

1. The "authorities", in particular the US government (and also the French) considered encryption a state prerogative and wanted to ban the use by the general public of such a powerful technique as RSA. This was hardly possible, but the export of RSA technology from the USA was banned - to the considerable loss of some US companies. RSA software was developed in many other countries, including Ireland (Baltimore Technologies) from the early 1980's, and sold worldwide free from US competition.

2. The size of the integers (512, 1024, 2048 bits) required for RSA to be secure, and its inherent cubic complexity, made implementations in silicon technology difficult while software implementations were slow. The invention of the sub-exponential techniques of factorising and solving the DL problem (using factor bases (see Chapter 9)) led to a search for other technologies, notably Elliptic Curves, immune to such attacks and capable of providing equal security with smaller numbers, and hence faster.

3. A craze for patenting algorithms, not least RSA itself, further led to searches for other approaches to avoid these patents, and possibly establish new ones. It is questionable if such patents could stand up in a court of law, but their existence (even granted to algorithms long after they had entered the public domain) was a strong deterrent to the use of established technology and a spur to new invention. (This problem was highlighted later in the AES project, Chapter 2, for symmetric algorithms where participants were made to forswear all notins of patents.)

Two strands of all this creative activity will be discussed in this module:

1. The digital signature standard, DSA and its relations
2. Variants of Diffie Hellman

5.1 DSA and Related Algorithms

Digital Signature algorithms have been developed which do not have a mode in which they can be used for public key encryption. There are many examples such as Fiat-Shamir, and El Gamal (closely related to DSA). DSA itself seems to have had an almost identical counterpart invented - and not patented - in Germany.

The DSA algorithm is defined in the DS standard (FIPS 186) which includes an associated Hash Function.

Users of a DSA implementation share a common prime modulus p, and a base g. The order of g is q, with q a large prime dividing $(p-1)$.

A user has a secret key x and a public key $y = g^x \ mod \ p$. To sign message m, the signatory forms $H(m)$, a hash of the message (that is a digest of the message using a scrambling hash function, see Chapter 6, typically 160 bits wide) and invents an exponent $k < q$. The signatory then forms

$$\begin{aligned} \mathrm{r} &= (\mathrm{g}^k \ mod \ p) mod \ q \\ \mathrm{s} &= \mathrm{k}^{-1}(H(m) + xr) mod \ q \end{aligned}$$

(k^{-1} is formed $mod \ q$). The signature to m is then $(r, \ s)$. In this scheme k and x are hidden from discovery from r and y by the Discrete Logarithm (DL) problem.

To verify the signature a person must also form $H(m)$ from the message and evaluate $s^{-1} \ mod \ q$. The he tests if

$$r = ((g^{s^{-1}H(m)} * y^{s^{-1}r}) mod \ p) mod \ q$$

If he is using the correct public key y, and the message is unchanged this equation will hold because

$$\begin{aligned} \text{Right Hand Side} &= (\mathrm{g}^{s^{-1}(H(m)+xr)} mod\ p) mod\ q \\ &= \mathrm{g}^{k}\ mod\ p\ mod\ q\ =\ r \end{aligned}$$

Note that the secrecy of k protects that of x, and vice versa, in the equations. Note also that the same k should not be used twice by the same signatory. (If the same k is used twice then his x could be found from)

$$s_1^{-1}(H(m_1) + xr_1) = s_2^{-1}(H(m_2) + xr_2)$$

because $(r_1,\ s_1)$ and $(r_2,\ s_2)$ and m_1 and m_2 are known.

Note also that both signatory and verifier have to perform (slow) modular inversions for s^{-1} and k^{-1}.

A more serious weakness is that a fraudster could invent any pair $(r,\ s)$ - or use somebody else's signature to message m - and form $y = (r^s g^{-H(m)})^{r^{-1}}$ and issue it claiming y to be his public key and $(r,\ s)$ his own signature to m. It would pass the verification test and the fraudster could lay claim to message m without ever having a secret key x. This attack would only work if the fraudster's "public key, y" was uncorroborated by anyone, for example uncertified.

5.2 The Korean Digital Signature Algorithm (KDSA)

KDSA addresses this last problem by building the signatory's certificate, z, into the hash $H(z|m)$, thus making the fraudster's formation of y above infeasible.

Other modifications include the avoidance of the need to find modular inverses at every signing and verification; and the use of $\oplus$ (XOR) and also the Hash, to foul up the arithmetic and block other inversions.

As before we have secret key $x < q$, but the public key $y = g^{x^{-1}}\ mod\ p$. To sign, the signatory invents k and forms

$$\begin{aligned} \mathrm{r} &= \mathrm{H}(\mathrm{g}^{k}\ mod\ p) \text{ so that } r < q \\ \mathrm{s} &= \mathrm{x}(\mathrm{k}\text{-}(\mathrm{r} \oplus H(z|m)))mod\ q \end{aligned}$$

$(r,\ s)$ is the signature to m by the holder of the certificate z.

The verifier forms $e = r \oplus H(z|m)$, the assumption being that the signatory's certificate z accompanies message and signature.

Test: Is $r = H(y^s * g^e\ mod\ p)$?

If the signature is correct and the message unchanged the

$$\text{Right Hand Side} = \begin{array}{l} \mathrm{H}(\mathrm{g}^{x^{-1}s+r\oplus H(z|m)} mod\ p) \\ \mathrm{H}(\mathrm{g}^{k}\ mod\ p) = r \end{array}$$

5.3 Variants of Williamson /Diffie-Hellman Shared Secret

The original scheme is as follows: A, B share a modulus p (prime) and a base α. $Order(\alpha) = q = large\ prime$.

Objective: To establish a shared secret (SS)

A		**B**
$Invents$ x		$Invents$ y
$Forms\ \alpha^x\ mod\ p$ and sends to B		$Forms\ \alpha^y\ mod\ p$ and sends to A
	↘ ↙ ↙ ↘	
$Forms\ (\alpha^y)^x\ mod\ p$		$Forms\ (\alpha^x)^y\ mod\ p$
	SS=α^{xy}	

Note 1: A and B have no proof as to the other with whom they share the secret.

Note 2: Suppose $Order(\alpha) = j * k$, then an intruder on the communication link between A and B could turn α^x into α^{xj}, and α^y into α^{yj} so the shared secret would be $SS = \alpha^{xyj}$. There would then be only k possible secrets, which could be exhaustively tried by the intruder. Thus $Order(\alpha)$ must be a large prime.

The following Japanese variants introduce the use of certified public keys for A and B - i.e. certificates. They enable A and B to assure themselves as to the identity of the other with whom they share SS.

5.4 MTI/CO

A has a certified public key $\alpha^x\ mod\ p$, x being A's secret key. B has a certified public key $\alpha^y\ mod\ p$, y being B's secret key. A then uses B's certificate and B uses A's certificate - got either from some public data base, or sent by its owner to the other, as follows:

A		**B**
Gets α^y		*Gets* α^x
Invents r		*Invents* s
Forms α^{ry} and sends to B		*Forms* α^{sx} and sends to A
	$\searrow\swarrow$	
Forms $((\alpha^{sx})^{x^{-1}})^r$ *mod* p		*Forms* $((\alpha^{ry})^{y^{-1}})^s$ *mod* p
	SS=α^{sr}	

Note 1: $x^{-1},\ y^{-1}$ are found *mod* q $\ q = Order(\alpha) = large\ prime$ (as above)

Note 2: A fraudster E could deceive B into believing that he (B) is talking with E, when he is really talking with A, as follows:

E forms $(\alpha^x)^e$ *mod* p from $A's$ public key and get B to believe this is really $E's$ public key, then B would send α^{sxe} supposedly to E but E gets it routed to A. A sends α^{ry} to B, but E intervenes en route and turns it into α^{rye} which he sends on to B. Now A and B proceed as though nothing had happened but the $SS = \alpha^{sre}$ *mod* p, and whereas A knows he is talking to B, B thinks he is talking to E.

This fraud is called "Unknown Key Share". It could be used by E as follows: E knows A is to send some secret to B. Using the fraud B receives the secret but think it is from E. E can hopefully find out the secret from B, because B will not be reluctant to discuss it with E, whom he thinks knows the secret already. Typically SS is a symmetric key for a subsequent exchange of information.

5.5 MTI/AO

$A,\ B$ have certificate $\alpha^x,\ \alpha^y$ as in MTI/CO. This time we suppose they send these to the other together with $\alpha^r,\ \alpha^s$ as follows:

A		**B**
Has α^x (Cert)		*Has* α^y (Cert)
Invents r		*Invents* s
Forms α^r and sends both to B		*Forms* α^s and sends both to A
	$\searrow\swarrow$	
Forms $(\alpha^y)^r.(\alpha^s)^x$		*Forms* $(\alpha^x)^s(\alpha^r)^y$
	SS=α^{xs+yr} *mod* p	

Note 1: Unknown key share is again possible if the fraudster E manages

to get a public key α^{ex} and impersonate A at the start, and also change α^s to α^{es} in B's transmission to A. As before A and B will end up sharing $SS = \alpha^{exs+yr}$ with A knowing he talks with B, but B thinking he talks with E.

Note 2: "Unknown Key Share" is not possible if users check that other people's public keys are properly certified by a CA (or Certification Authority) and the CA makes sure that all the owners who apply for their public keys to be certified actually do own the corresponding secret key. (In the above fraud E does not know (ex)).

In the above MTI/AO case there is no future-proofing. That is, should secret key exponents $x,\ y$ become revealed sometime in the future then the SS can be reconstructed (and if it were itself a symmetric secret key, the messages protected by it are now at risk) as long as a record of the exchanges is available.

To counter this danger we use a third Canadian variant, MQV.

5.6 MQV

A has public key $u = \alpha^x$, secret key x.
B has public key $v = \alpha^y$, secret key y.

A		**B**
Invents x′		*Invents* y′
Forms u′ $= \alpha^{x'}$ *mod p*		*Forms* v′ $= \alpha^{y'}$ *mod p*
Send u, u′ to B		*Sends* v, v′ to A
	↘ ↙ ↙ ↘	
Forms r=u′$x + x'$		*Forms* s=v′$y + y'$
Forms $(\mathrm{v}^{v'}.v')^r$		*Forms* $(\mathrm{u}^{u'}.u')^s$
$=(\alpha^{v'y+y'})^r$		$=(\alpha^{u'x+x'})^s$

SS=$\alpha^{(v'y+y')(u'x+x')}$ *mod p*

Note 1: Unknown key share is not possible. Why?
Note 2: Future-proofing is assured because even if $x,\ y$ are discovered, x' and y' are still secret and SS cannot be reconstructed without them.

5.7 Rabin Encryption (An example of non-RSA public key encryption)

Beside digital signatures which do not offer public key encryption, there are many other PKC techniques which do provide encryption. An example is

due to Rabin.

Rabin encryption relies on Quadratic Residues (QRs). A QR *mod* p (prime) is an integer $y = x^2 \ mod \ p$ for some x. A Quadratic Non-Residue is an integer which is not such a square.

Example mod 17

$$1^2 = 1,\ 2^2 = 4,\ 3^2 = 9,\ 4^2 = 16,\ 5^2 = 8$$
$$6^2 = 2,\ 7^2 = 15,\ 8^2 = 13,\ 9^2 = 13,\ 10^2 = 15, 11^2 = 2,\ etcetra$$
QRs are 1, 2, 4, 8, 9, 13, 15, 16
QNRs are 3, 5, 6, 7, 10, 11, 12, 14
Note: $11^2 = 6^2 = (-11)^2$ of course etc.

Now if p is prime there exists a primitive α such that every element *mod* p is a power of α, α^i (see Chapter 3).

If $\beta = (\alpha^i)^2$ for some i, so that β is a QR, then $\beta^{\frac{p-1}{2}} = \alpha^{(p-1)i} = 1 \ mod \ p$. Conversely if $\beta = \alpha^{2i+1}$ then $\beta^{\frac{p-1}{2}} = \alpha^{(p-1)i} * \alpha^{\frac{p-1}{2}} = 1 * (-1) = -1$ because there is only one other square root of 1.

So $\beta^{\frac{p-1}{2}} = 1$ implies β is a QR and $\beta^{\frac{p-1}{2}} = -1$ implies β is a QNR.

If the modulus n is the product of two prime $p,\ q$ then only a quarter of the integers *mod* n are QRs, and each has four square roots - found by finding the square roots *mod* p and *mod* q (two each) and putting them together using the Chinese Remainder Theorem (CRT).

Rabin encryption uses $n = p.q$ as modulus and encryption of message m is $c = m^2 \ mod \ n$.

Decryption is $c^{\frac{1}{2}} \ mod \ p = \pm \ m \ mod \ p$ and $c^{\frac{1}{2}} \ mod \ q = \pm \ m \ mod \ q$ and m can be found using the CRT. Obviously the security is based on the infeasibility of an attacker factorising n.

Note that if the solution *mod* p and *mod* q to $c^{\frac{1}{2}}$ are $\pm m_1$ and $\pm m_2$ respectively our four solutions for m are formed by :

$$(\mathrm{m}_1,\ m_2) \rightarrow m \quad (\mathrm{m}_1,\ -m_2) \rightarrow m'$$
$$(\text{-}\mathrm{m}_1,\ m_2) \rightarrow -m' \quad (\text{-}\mathrm{m}_1,\ -m_2) \rightarrow -m$$

Example *mod* 35 $m = 13\ m^2 = 29$
$m_1^2 \ mod \ 5 = 4 \quad m_1 = \pm 2 \ mod \ 5 = 2,\ 3$
$m_2^2 \ mod \ 7 = 1 \quad m_2 = \pm 1 \ mod \ 7 = 1,\ 6$
So CRT gives $2,\ 1 \rightarrow 22, \quad 2,\ 6 \rightarrow 27, \quad 3,\ 6 \rightarrow 13, \quad 3,\ 1 \rightarrow 8$
$22^2 = 13^2 = 27^2 = 8^2 = 29 \ mod \ 35$

Note: $(22^2 = 27^2 \; mod \; 35) \rightarrow (22-27)(22+27) = 0 \; mod \; 35$ or $(-5)(49) = 0 \; mod \; 35$ or $(-5)(14) = 0 \; mod \; 35$. That is if we know the four square roots we can factorise the modulus.

For Rabin decryption the four square roots must not be revealed to outsiders (remembering that an attacker who encrypts m already knows two of the square roots). Therefore the intended recipient must be able to select the correct square root of c easily without revealing the others. One method is to attach to m some check sum which is validated on decryption and if correct then m (and not $-m, \; m'$ or $-m'$) is produced by the validation procedure..

If $p = 4k + 3$ for some $k, \; c^2 = c^{p+1} = c^{4k+4} = mod \; p$. So $c^{\frac{1}{2}} = c^{k+1}$.

Example $p = 23, \; c = 2, \; c^{\frac{1}{2}} = 2^{5+1} = \pm 18 \; mod \; 23 = 18, \; 5$

But if $p = 4k + 1$ a more sophisticated method is needed to extract square roots $mod \; p$.

5.8 Extracting Square Roots $mod \; p$

To find a' such that $(a\prime)^2 = a \; mod \; p$ where $a, \; p$ given. (Assuming a is a QR i.e. $a^{\frac{p-1}{2}} = +1$).
a' is a solution to

$$\mathrm{x}^{p+1} = a \; mod \; p \tag{0.1}$$

i.e. $(x^{\frac{p+1}{2}} - a')(x^{\frac{p+1}{2}} + a') \; mod \; p$ has roots a' and $-a'$.

Consider $f(x) = x^2 - bx + a$ over GF(p) with $f(x)$ irreducible. i.e. $w^2 = (b^2 - 4a)$ is a QNR in GF(p) so $(w^2)^{\frac{p-1}{2}} = -1 \rightarrow w^p = -w$.

The roots of $f(x)$ are $\alpha = r + sw$ and $\beta = r - sw$ with $r = \frac{b}{2}, \; s = \frac{1}{2}$.

Now α is a root of equation **??** because

$$\begin{aligned} \alpha^p = (r + sw)^p \quad &= \mathrm{r}^p + s^p w^p \; mod \; p \\ &= \text{r-sw} = \beta \\ Therefore \; \alpha^{p+1} \quad &= \alpha\beta \; = \; a \text{ from } f(x) \\ Therefore \quad &\alpha \text{ satisfies equation } \mathbf{??}. \end{aligned}$$

Similarly β is a root of equation **??** because

$$\begin{aligned} \beta^p = (r - sw)^p &= \mathrm{r}^p - s^p w^p \ mod\ p \\ &= \mathrm{r{+}sw} = \alpha \\ Therefore\ \beta^{p+1} &= \alpha\beta = a \text{ from } f(x) \\ Therefore &\ \beta \text{ satisfies equation } \mathbf{??}. \end{aligned}$$

Therefore $f(x)|(x^{p+1} - a)$ so $f(x)|(x^{\frac{p+1}{2}} - a')(x^{\frac{p+1}{2}} + a')$. But $f(x)$ is irreducible, therefore $f(x)|(x^{\frac{p+1}{2}} - a')$ or $f(x)|(x^{\frac{p+1}{2}} + a')$. Therefore to find a', divide $x^{\frac{p+1}{2}}$ by $f(a)$ and look at the remainder.

Example *Find* $2^{\frac{1}{2}}$ *mod* 17
Consider $(x^{18} - 2)$ *mod* 17. $2^{\frac{1}{2}}$ *is a root.*
Form $f(x) = x^2 - bx + 2$. $f(x) = x^2 - x + 2$

$$\begin{aligned} Because\ : \mathrm{w}^2 &= \mathrm{b}^2 - 8 \text{ is a QNR} \\ Try\ \mathrm{b{=}1}.\ \mathrm{w}^2 &= \text{-7} = 10 \\ 10^{\frac{p-1}{2}} &= 10^8 = ? \\ 10^8 &= \text{-1 mod 17} \\ Therefore &\ \mathrm{w}\ is\ a\ QNR \end{aligned}$$

So take $b = 1$. Divide into $x^{\frac{p+1}{2}} = x^9$. $(x^2 - x + 2)/x^9 = (x^7 + x^6 - x^5 - 3x^4 - x^3 + 5x^2 + 7x - 3)$ *with remainder* 6 *mod* 17.
So $a' = 6$ $(or(-6))$. Check $6^2 = 36 = 2$ *mod* 17