Authenticated ID-based Key Exchange and remote log-in with simple token and PIN number

Michael Scott

School of Computer Applications
Dublin City University
Ballymun, Dublin 9, Ireland.
mike@computing.dcu.ie

Abstract. Authenticated key exchange protocols tend to be either token based or password based. Token based schemes are often based on expensive (and irreplaceable) smart-card tokens, while password-only schemes require that a unique password is shared between every pair of correspondents. The magnetic strip swipe card and associated PIN number is a familiar and convenient format that motivates a combined "two-factor" approach. Finally we suggest an extension of the scheme for use in a client-server scenario.
Keywords: Authenticated key exchange, pairing-based cryptosystems.

1 Introduction

Consider a small group of people who wish to communicate in private. It is inconvenient for them all to meet, so their point of commonality is access to a Trusted Authority (TA). These people expect to be able to trust one another – why else would they want to communicate in secret?

The trusted authority issues to each member of the group an *individual secret* number associated with their identity. Each member then "splits" this number into a small user-selected PIN number, and a larger number which is stored in the hardware token. The *individual secret* can be reconstructed from these two components whenever it is required in the key-exchange protocol. The hardware token is in some convenient form-factor, but can easily be copied (which is convenient as it allows backup copies to be created).

To communicate, each inserts their own token into a computer, types in their memorised PIN number and the identity of their correspondent. The *individual secret* is reconstructed, an ephemeral session key is calculated and the pair communicate in privacy.

Note that each participants PIN is a secret unto themselves alone. This therefore differs from schemes like SPEKE [11], where a different low entropy secret must be shared between every pair of users, which could lead to a multiplicity of PINs to be remembered.

Such a scheme as we propose is clearly open to the following active "insider" attack. Bob steals or copies Alice's token, takes it home and carries out multiple off-line key exchange calculations trying every possible number for Alice's PIN. Eventually he will find it.

The scheme as proposed is also vulnerable to identity theft. If Trent should rob Bob's token and rubber-hose his password, then Trent can become Bob at will, and can launch "insider" attacks. While we can assume that a token can be secretly copied, it can also be assumed that PIN theft cannot be carried out without the knowledge of the legitimate PIN-holder. However the scheme should be able to withstand an outsider stealing or copying any number of tokens and/or any number of passwords as long as none of the "stolen" passwords is associated with any of the stolen tokens. This seems to be the best that can be expected from such a scheme.

We recognise that, in common with schemes like SPEKE, ours is vulnerable to an entity Trent who steals or copies a token, and is allowed to attempt multiple on-line guesses of the associated PIN while attempting to form a connection with legitimate users. It is assumed that suitable measures would be put in place to detect multiple failed connections from a purported individual, and to subsequently exclude him or her. We recognise however that in this context such guessing attacks could be "spread around" and tried against many legitimate individuals. For this reason the PIN number might be say 8 digits in length, rather than the more common 4-digits.

Finally we would like our scheme to have the property of "forward secrecy". If at some time time in the future all long-term secrets are revealed, including those of the trusted authority, previously recorded ciphertext should remain inviolable.

2 The proposed scheme

The key exchange algorithm reconstructs the individual secret from the large number stored on the hardware token and the PIN. It is important that a correct individual secret cannot be detected from its format – it is just a number, and any number is possible. Essentially the relationship is linear and of the form $D = N + PIN$, where D is the individual secret, and N is the number on the token.

In our proposed scheme the trusted authority knows a master secret number s. User identities are mapped using a suitable hash function to points on a particular elliptic curve. The user Alice has her identity hashed and mapped to a point A of large prime order on the curve, and chooses a desired PIN number α. After authenticating herself to the trusted authority she receives A and sA, calculates αA, subtracts the two, stores A and $(s-\alpha)A$, and memorises α. As in a simple secret sharing scheme these two halves need to be reunited to reconstruct the correct value sA. Clearly Alice cannot determine s without solving a difficult discrete logarithm problem. We have the simple linear relationship $sA = (s - \alpha)A + \alpha A$.

Note that if an increment is added to α, the effect can be offset by incrementing s by the same amount. So for example if α is incremented by δ, the relationship $(s+\delta)A = (s-\alpha)A + (\alpha+\delta)A$ holds. This implies that if two or more correspondents increment their PINs by the same amount, they can still complete the key agreement process, although this time the key will be a function of $(s+\delta)$. This in turn implies that an opponent who has stolen multiple tokens cannot determine individual PINs – he cannot distinguish the effect of $(\alpha+\delta)$ from α. So he can determine the difference between PINs, but not actual PINs, which is useless if he should attempt to communicate with a legitimate party.

First let us consider the implementation of such a protocol using standard public key cryptographic constructions. One common approach is for a trusted centre to digitally "sign" our identity [9], [16]. This secret signature (our *individual secret*) is subsequently used in the key exchange. Unless each correspondent's signature is valid, a mutual session key cannot be negotiated. The trusted authorities public key is cleverly integrated into the key exchange to ensure that this is true.

Now if every participant is to have their own PIN number, then it would appear that this PIN must also be generated by the signing process. A simple approach would be to remove part of the signature for use as a PIN. Unless the removed part were "put back in" the exchange could not take place successfully. However this will not satisfy our security requirement. An attacker who captures a token can exhaustively search through all possible PINs until the complete signature can be verified by the TA's public key.

3 The Tate pairing

The Weil and Tate pairings are operations on pairs of points on an elliptic curve. See [14] for some early applications, specifically in breaking elliptic curve based cryptosystems based on certain weak curves, by reducing the ECDLP problem to a DLP problem over an extension of the base field. More recently these pairings have been found to have applications for "good". For example Joux's clever tripartite Diffie-Hellman key exchange [12], Boneh and Franklin's ID-based encryption scheme [4], and the BLS short signature scheme [5].

The Tate pairing [8] is written as $e_r(P,Q)$, where P and Q are both points on a curve defined over $\mathbb{F}_{p^k}$, and P is of order r. Note however that there is no requirement for Q to be of order r [17]. The Tate pairing evaluates as an element of order r in the field $\mathbb{F}_{p^k}$. In a normal EC setting it is quite unexciting; if P and Q are linearly dependent, that is if the curve group structure is cyclic, it evaluates as 1. The trick is to use points on P and Q which lie in different subgroups, so that $P \neq sQ$ for any value of s. This can only happen on a curve with a non-cyclic group structure. This behaviour is possible, and convenient, if $r \mid p^k - 1$ and k is small. But note that the elliptic curve discrete logarithm problem on a curve with a small k value is readily reduced to a discrete logarithm problem in the field $\mathbb{F}_{p^k}$, and so parameter sizes must be adjusted upwards to take account of this.

These rather restrictive conditions are met by certain non-supersingular curves. In practise it is not difficult to find non-supersingular curves with moderate values of k, for example the MNT curves described in [15], or using the methods described in [1], [3], [6], or [19]. If $k > 1$ (which we will assume here) a convenient way to make sure that P and Q are in seperate subgroups is to ensure that P's coordinates are in $\mathbb{F}_p$ while Q's coordinates are in $\mathbb{F}_{p^k}$. In fact as pointed out in [2] when k is even Q can be conveniently placed on the "twisted" curve over $\mathbb{F}_{p^{k/2}}$. This also speeds up the calculation [17].

The Tate pairing has the following useful properties.

1. $e_r(aP, bQ) = e_r(P, Q)^{ab}$ for all $a, b \in \mathbb{F}_r$ (Bilinearity)
2. $e_r(P, P) = 1$

A modified pairing works well with supersingular curves, for which k is always guaranteed to be small, and this is the case we will look at first.

Consider the following supersingular elliptic curve over the field $\mathbb{F}_p$, where $p \equiv 3 \bmod 4$ is a 512-bit prime:

$$E(\mathbb{F}_p) : y^2 = x^3 + x$$

The order of this supersingular curve is $q+1$, and $k = 2$ as $r \mid p+1 \mid p^2-1$. The discrete logarithm problem on this curve can be reduced to a discrete logarithm problem in the field $\mathbb{F}_{p^2}$, but this is still considered intractable for a prime of the specified size.

On this supersingular curve, using the terminology of Verheul [20], we can define a *distortion map* $\phi : E(\mathbb{F}_p) \to E(\mathbb{F}_{p^2})$ which maps a point (x, y) to the point $(-x, iy)$ in a different subgroup, where $i^2 = -1$. Note that the condition $p \equiv 3 \bmod 4$ implies that i lies not in $\mathbb{F}_p$, but rather in the quadratic extension field, so the transformed point lies on the same curve defined over $\mathbb{F}_{p^2}$.

Now consider the *distorted Tate pairing*:

$$\hat{e}_r(P, Q) = e_r(P, \phi(Q)) = \mu$$

where P (of order r) and Q are points on the curve, $\mu \in \mathbb{F}_{p^2}$ is of order r, and $e_r(P, Q)$ is the Tate pairing [8]. Under these circumstances the distorted Tate pairing will produce a non-trivial result ($\hat{e}_r(P, Q) \neq 1$). This pairing has the following properies:

1. $\hat{e}_r(aP, bQ) = \hat{e}_r(P, Q)^{ab}$ for all $a, b \in \mathbb{F}_r$ (Bilinearity)
2. $\hat{e}_r(P, P) \neq 1$
3. $\hat{e}_r(P, Q) = \hat{e}_r(Q, P)$

4 The Key Exchange Protocol

The trusted authority generates a 512-bit prime $p \equiv 3 \bmod 4$ such that $p + 1 = c.r$, where r is a 160-bit prime, and chooses a suitable hash function $H : \{0,1\}^* \rightarrow E(\mathbb{F}_p)$ which hashes identity strings to a point on the supersingular curve. It publishes the public parameters $\{p, r, H\}$. The trusted authority then generates its own secret 160-bit number $s \in \mathbb{F}_r$.

For individuals to register with the trusted authority they must prove their identity ID. In return they are supplied with I and their individual secret sI where $I = c.H(ID)$, a point of order r on the curve. They store on their hardware token I and $(s - \rho)I$, where ρ is their personal choice of PIN number.

The protocol is illustrated in Table 1. To a passive eavesdropper this appears very like an instance of the Diffie-Hellman key exchange in $\mathbb{F}_{p^2}$. It would also appear that anyone not knowing, and not able to construct from available information, $\hat{e}_r(A, B)^s$ or equivalently sA or sB, is not in a position to launch a classical man-in-the-middle attack against it. Another way of looking at it is as a SPEKE-like key exchange, where the group generator is only known to the two legitimate participants. This is an established technique in the process of standardisation [10], and no active attacks are known against it.

Alice	**Bob**
Generates random $a < r$	Generates random $b < r$
$ID_a \rightarrow$	$\leftarrow ID_b$
$B = H(ID_b)$	$A = H(ID_a)$
$P_a = \hat{e}_r((s - \alpha)A + \alpha A, B)^a$	$P_b = \hat{e}_r((s - \beta)B + \beta B, A)^b$
$P_a \rightarrow$	$\leftarrow P_b$
If $P_b \leq 1$ or ${P_b}^r \neq 1$ Abort	If $P_a \leq 1$ or ${P_a}^r \neq 1$ Abort
Key $= {P_b}^a = \hat{e}_r(A, B)^{sab}$	Key $= {P_a}^b = \hat{e}_r(B, A)^{sab}$

Observe that the fact that Alice can easily obtain Bob's password if she has a copy of his token can be blamed on the relationship $\hat{e}_r(sA, B) = \hat{e}_r(A, sB) \neq 1$, which is clearly required by the protocol. As pointed out before it is now a simple matter to search among δ until $\hat{e}_r(sA, B) = \hat{e}_r((s - \beta)B + \delta B, A)$, in which case $\delta = \beta$. The underlying reason that this is possible is due to the fact that the Decisional Diffie-Hellman (DDH) problem is easy in this setting [4], and hence we can distinguish the action of a "right" password guess from a wrong one.

The fact that $\hat{e}_r(sA, B) \neq 1$ depends directly on the property $\hat{e}_r(P, P) \neq 1$. Since A and B are members of the same subgroup, we know that $A = uP$ and $B = vP$ for some u, v and P. Therefore $\hat{e}_r(sA, B) = \hat{e}_r(suP, vP) = \hat{e}_r(P, P)^{suv}$.

4.1 Security Analysis

First we demonstrate that anyone who can determine the key from publicly transmitted information is in a position to solve an instance of the Bilinear Diffie-

Hellman problem (BDH) [4], which is believed to be hard. The BDH problem can be described succintly thus: Given $\{P, sP, aP, bP\}$, determine $\hat{e}_r(P,P)^{sab}$.

Assume an Oracle exists which when input the exchanged public data $\{A, B, P_a = \hat{e}_r(sA, aB), P_b = \hat{e}_r(bA, sB)\}$ outputs the key $\hat{e}_r(A,B)^{sab}$.

Now we will demonstrate how this oracle can be used to solve an instance of the BDH problem. Generate a random u and present to the Oracle $\{P, uP, \hat{e}_r(sP, u(aP)), \hat{e}_r(bP, u(sP))\}$ which is the same as $\{P, uP, \hat{e}_r(sP, a(uP)), \hat{e}_r(bP, s(uP))\}$. The oracle will respond with $\hat{e}_r(P, uP)^{sab} = \hat{e}_r(P,P)^{sabu}$. We recover $\hat{e}_r(P,P)^{sab}$ by raising the output of the Oracle to the power of $1/u$.

Note that the inverse is not necessarily true: It cannot be concluded that a passive eavesdropper could use a BDH oracle to discover the key. This is because, although A and B are from the same subgroup, and therefore $A = xB$ for some value of x, this x value will not be known due to the action of the hash function.

However a third group member who can solve BDH, can easily break the authentication of the scheme, as pointed out by Michael Cheng [7]. Imagine a third member X of the group has X and sX. Now $A = wX$ for some w and $B = zX$ for some z, so if X can solve BDH with input $\{X, sX, wX, zX\}$ then he can find $\hat{e}_r(X,X)^{swz} = \hat{e}_r(wX, zX)^s = \hat{e}_r(sA, B)$.

The inverse is also true – an oracle that can determine $\hat{e}_r(sA, B)$ from $\{X, sX, A, B\}$ can be used to solve an instance of BDH: Supply as input $\{P, sP, aP, bP\}$ amd the output will be $\hat{e}_r(P,P)^{sab}$.

The sixth step of the protocol checks the order of the value received from the other party. This is to avoid a small subgroup confinement attack, whereby an active attacker substitutes a value of small order. Such a value when exponentiated remains in the same small subgroup, and can be found by a short exhaustive search through all the possible subgroup members. This is a well-known attack on Diffie-Hellman-like protocols, and the standard response is to check, as shown above, that the received value is indeed in the subgroup of order r. However the extra work involved can be avoided by using the technique of Lim-Lee [13]. The idea is simple – ensure that there are no small subgroups, and if there are (for example there may be an unavoidable subgroup of order 2), then explicitly check for such cases. This places an extra constraint on parameter generation, however in practise this is not a problem. It is however necessary that the idea, originally described in the $\mathbb{F}_p$ setting be adapted for the field $\mathbb{F}_{p^2}$ which has $p^2 - 1$ elements. Briefly, ensure by design that the modulus p is of the form such that $p + 1 = 2rq$, where q is also prime. Then process a received value as follows: Raise it to the power of $p - 1$ using the frobenius action – this requires a single modular division [17] – and then square it. The resulting value will either be 1, in which case the protocol should be aborted, or it will be a point of large order, whose order divides $r.q$.

Finally we maintain that this protocol has the property of forward secrecy. The ephemeral values a and b are discarded after use, and without them the agreed key cannot be determined in the future by anyone.

5 A Client-Server Protocol

In a client-server context multiple clients are assumed to wish to access an individual server. In this context the security requirements are rather different. Although the server may trust each client individually, it is not at all clear that clients may trust one-another. Therefore in this context it is important that the "insider" attacks cannot succeed, that is if Alice obtains Bob's token, she cannot determine his password.

The idea is to eliminate this attack by exploiting the fact that clients only need to communicate with servers, not with one-another. This is acheived by putting all the clients together in one sub-group, with the server in another. We want a setting where there is no bilinear pairing between points in the same subgroup that yields a non-trivial result. This implies a setting where the conventional Tate pairing satisfies $e_r(P, P) = 1$ and where no distortion map exists. This will be the case for non-supersingular curves where $k > 1$. See [8] section 2.4.

The first step then is to find a non-supersingular curve with low, even k value (say $k = 2$), and for which no distortion map exists. A suitable example $k = 2$ curve is given in [17], and it is our experience that it is not difficult to find such curves, using for example the "folklore" method described in [3]. Such curves do not support a distortion map – see theorem 4.1 of [20].

Assume for simplicity that the server takes on itself the role of Trusted Authority. It generates a $k = 2$ non-supersingular curve over $\mathbb{F}_p$, where p is 512-bits, such that the curve order is $c.r$ and is divisible by the 160-bit prime r. It also generates its own secret $s \in \mathbb{F}_r$. It is assumed that the server does not require a password. If the server's identity is ID_s, then this is mapped to a point S on the same curve, but this time over the extension field $\mathbb{F}_{p^2}$, using a second hash function $H_2 : \{0, 1\}^* \rightarrow E(\mathbb{F}_{p^2})$, such that $S = H_2(ID_s)$. So S is a point on the curve $E(\mathbb{F}_{p^2})$. The server issues the curve parameters and the H_2 hash function.

Alice registers with the server exactly as before and is issued with A and an individual secret of the form sA, where $A = c.H(ID_a)$ is a point of order r on the base elliptic curve $E(\mathbb{F}_p)$. As before she partitions her individual secret up between the hardware token, and a short memorised PIN. Note that here we use the unmodified Tate pairing.

To log-in, Alice executes the following protocol.

Alice	**Server**
Generates random $a < r$	Generates random $x < r$
$ID_a \rightarrow$	$\leftarrow ID_s$
$S = H_2(ID_s)$	$A = c.H(ID_a)$
$P_a = e_r((s - \alpha)A + \alpha A, S)^a$	$P_s = e_r(A, sS)^x$
$P_a \rightarrow$	$\leftarrow P_s$
If $P_s \leq 1$ or ${P_s}^r \neq 1$ Abort	If $P_a \leq 1$ or ${P_a}^r \neq 1$ Abort
Key $= {P_s}^a = e_r(A, S)^{sax}$	Key $= {P_a}^x = e_r(A, S)^{sax}$

In this case the insider attack does not work, as for example between Alice and Bob $e_r(A, B) = 1$. Note that there is a co-DDH problem which is easy to solve in this setting, but the standard DDH problem remains hard.

5.1 Security Analysis

Assume now that an oracle exists which when input the exchanged public data $\{A, S, P_a = e_r(sA, aS), P_b = e_r(bA, sS)\}$ outputs the agreed key. Then such an oracle can be used to solve an instance of the co-bilinear Diffie-Hellman (co-BDH) problem [4], which is also believed to be hard. The co-BDH assumption is that given $\{P, sP, aP, Q, sQ, bQ\}$ it is hard to calculate $e_r(P, Q)^{sab}$. Input $\{P, Q, e_r(sP, bQ), e_r(aP, sQ)\}$ to our oracle and it will respond with $e_r(P, Q)^{sab}$.

6 Implementational issues

In both protocols we have made careful use of the observation from [17] that the second parameter to the Tate pairing does not have to be of order r. This leads to considerable savings, as co-factor point multiplication (by c) can be very expensive. In the key exchange protocol no co-factor multiplication is required. For the client-server protocol, the server is required to perform one co-factor multiplication. However the calculation on the client side remains relatively lightweight.

Both protocols can also profit by compressing the pairing output [18]. For the case of $k = 2$ as suggested here, the pairing value can be compressed to one-half of its normal length – a single $\mathbb{F}_p$ value suffices instead of a full element of $\mathbb{F}_{p^2}$. See [18] for details. This has immediate and obvious bandwidth benefits. It also makes the pairing calculation a little faster [17].

7 Conclusions

Most pairing-based protocols work equally well over both supersingular and non-supersingular curves, for example Boneh and Franklin IBE [4]. Here we demonstrate for the first time that the differences between the two types of pairing can be exploited to create protocols with quite different properties. This in turn can be directly related to the difficulty (or non-difficulty) of solving the DDH problem.

The client-server protocol in particular may find useful application. The user password is remembered by the client and no vulnerable password file is stored on the server. The server's master key s is only needed for client registration purposes. For day-to-day use the server only needs access to sS.

References

1. P. S. L. M. Barreto, B. Lynn, and M. Scott. Constructing elliptic curves with prescribed embedding degrees. In *Security in Communication Networks – SCN'2002*,

volume 2576 of *Lecture Notes in Computer Science*, pages 263–273. Springer-Verlag, 2002.
2. Paulo S. L. M. Barreto, B. Lynn, and M. Scott. On the selection of pairing-friendly groups. In *Selected Areas in Cryptography – SAC'2003*, volume 3006 of *Lecture Notes in Computer Science*, pages 17–25, Ottawa, Canada, 2003. Springer-Verlag.
3. I. Blake, G. Seroussi, and N. Smart. *Advances in Elliptic Curve Cryptography*, volume 2. Cambridge University Press, 2005.
4. D. Boneh and M. Franklin. Identity-based encryption from the Weil pairing. *SIAM Journal of Computing*, 32(3):586–615, 2003.
5. D. Boneh, B. Lynn, and H. Shacham. Short signatures from the Weil pairing. In *Advances in Cryptology – Asiacrypt'2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 514–532. Springer-Verlag, 2002.
6. F. Brezing and A. Weng. Elliptic curves suitable for pairing based cryptography. Cryptology ePrint Archive, Report 2003/143, 2003. Available from `http://55 3 ug 5 2emm 4. o i ee ood.re t/2003/143`.
7. Michael Cheng. Personal communication, 2004.
8. S. Galbraith. Supersingular curves in cryptography. In *Advances in Cryptology – Asiacrypt'2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 495–513. Springer-Verlag, 2002.
9. Christoph G. Günther. An identity-based key-exchange protocol. In *EUROCRYPT '89*, volume 434 of *Lecture Notes in Computer Science*, pages 29–37, 1990.
10. IEEE Computer Society, New York, USA. *IEEE Standard Specifications for Password-Based Public-Key Cryptographic Techniques*, 2003.
11. D. Jablon. Strong password-only authenticated key exchange. *Computer Communication Review, ACM SIGCOMM*, 26:5–26, 1996.
12. A. Joux. A one-round protocol for tripartite Diffie-Hellman. In *Algorithm Number Theory Symposium – ANTS IV*, volume 1838 of *Lecture Notes in Computer Science*, pages 385–394. Springer-Verlag, 2000.
13. C.H. Lim and P.J. Lee. A key recovery attack on discrete log-based schemes using a prime order subgroup. In *Advances in Cryptology – Crypto '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 249–263. Springer-Verlag, 1997.
14. A. Menezes. *Elliptic Curve Public Key Cryptosystems*. Kluwer Academic Publishers, 1993.
15. A. Miyaji, M. Nakabayashi, and S. Takano. New explicit conditions of elliptic curve traces for FR-reduction. *IEICE Transactions on Fundamentals*, E84-A(5):1234–1243, 2001.
16. E. Okamoto. Proposal for identity-based key distribution systems. *Electronic Letters*, 22:1283–1284, 1986.
17. M. Scott. Computing the Tate pairing. To be presented Cryptographers Track RSA conference February 2005.
18. M. Scott and P. Barreto. Compressed pairings. In *Advances in Cryptology – Crypto 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 140–156. Springer-Verlag, 2004.
19. M. Scott and P. Barreto. Generating more mnt elliptic curves. Cryptology ePrint Archive, Report 2004/058, 2004. `http://55 3 ug 5 2emm 4. o i ee ood.re t/2004/058`.
20. E. Verheul. Evidence that xtr is more secure than super-singular elliptic curve cryptosystems. In *EUROCRYPT 2002*, volume 2045 of *Lecture Notes in Computer Science*, pages 195–210. Springer-Verlag, 2001.