An Efficient Scheme for Secure Message Transmission using Proxy-Signcryption

Chandana Gamage, Jussipekka Leiwo, and Yuliang Zheng

Peninsula School of Computing and Information Technology
Monash University, McMahons Road, Frankston, Vic 3199, Australia
{chandag,skylark,yuliang}@pscit.monash.edu.au

Abstract. Previous proxy signature schemes enable a principal to have a trusted proxy agent sign messages on its behalf. We present a proxy signature scheme that combines the functionality of proxy signing and encryption. This proxy-signcryption scheme is useful for applications that are based on unreliable datagram style network communication model where messages are individually signed and not serially linked via a session key to provide authenticity and integrity. Use of a proxy agent to perform signature function is desirable for applications that are expected to support computing devices with low computational power and storage capacities. Integration of encryption functionality to provide secrecy at no additional cost to the proxy signature generation is an efficient means by which to support the class of applications targeted by this research work such as e-commerce using mobile computing and communication devices.
Keywords: Digital Signatures, Proxy Signatures, Public Key Cryptography, Signcryption, Network Security, Secure Mobile Computing

1 Introduction

The large scale adoption of computing and network technologies for carrying out on-line transactions and message transmissions have been greatly supported by the research and development in the area of cryptography and network security. The cryptographic primitives or *tools* such as encryption or digital signatures are used to build protocols that provide specific security services such as transmission of a message over an insecure network while protecting the integrity and confidentiality of the message contents. The direct application of cryptographic primitives to build security services for tasks such as secure authenticated message transmission results in generic protocols that are widely applicable yet inefficient. Most often, the inefficiencies of a particular general purpose protocol can be reduced by customizing it to a specific application area. Alternatively, we can attempt to further improve the efficiency of underlying cryptographic primitives used to construct the security protocol. This approach is particularly interesting as most protocols use a combination of cryptographic primitives to provide a series of security services.

1.1 The Problem and Motivation

In this paper we consider the problem of secure and authenticated message transmission by a networked computer with low computational capacity. Many widely used personal communication devices such as digital assistants, hand held computers, pagers and mobile phones belong in this category. The low computational capability constraint is introduced to model the lack of hardware features in these devices to efficiently carry out the heavy mathematical computations required by certain cryptographic primitives such as digital signatures. We consider a very generic communication model for these low power computers in which a given device may transmit and receive message from an arbitrarily large number of other computers. This flexibility in communication rules out secure message transmission using straightforward symmetric cryptography which requires a shared secret between communicating parties. Therefore, we look at the use of asymmetric or public key cryptography for secure message transmission. Our work is motivated by the proliferation of low power devices mentioned above and the many emerging applications for these devices such as stock portfolio management, short message services and collection of sensitive field data which require secure and authenticated data transfers.

1.2 Research Contribution

We identify the digital signing of a message as the most computationally intensive cryptographic operation in secure message transmission. Therefore, we use a proxy signature scheme that allows off-loading of heavy computational work from a low power device to a more powerful server. While the original proxy signature scheme we use focus on secure delegation of signing rights to a trusted proxy agent, our focus is to further improve the efficiency of this cryptographic primitive by combining it with another cryptographic primitive, namely, encryption. Using this improved cryptographic primitive, which we call proxy-signcryption, we show its use in a protocol for the particular application area of short message transmission.

1.3 Structure of the Paper

In section 2.2 we look at the main conceptual schemes available for the secure use of a (potentially more powerful) proxy agent to digitally sign messages on behalf of an original signer (also called a principal). This provides the background for what we have initially set out to achieve, that is an efficient proxy signature scheme for low power computing devices to allow secure and authenticated message transmission. Thus, in section 3 we describe an implementation of a practical proxy signature scheme combining the original digital proxy signature scheme and the signcryption public key cryptographic primitive. In section 4 we analyze the performance of our proposed proxy signature implementation in terms of computational speed and bandwidth requirement for a standard communication model. In section 5 we present security arguments to establish that the combination of the two secure protocols in to a new protocol does not reduce the security of the resulting protocol. Also, we provide security arguments as to how the changes we have made to the original schemes makes the new proxy-signcryption scheme more practicable. In section 6 we summarize and conclude.

2 Related Work

2.1 Overview of High-Speed Digital Signature Schemes

A digital signature of a message is a publicly known value computed using a secret known only to the signer and the message content. The process of generating a digital signature is highly computational intensive due mainly to its use of modular exponentiations. Digital signatures are a major component of any secure communication infrastructure and transaction protocol. Therefore, devices that need to securely communicate must be capable of these computations. However, many new devices such as cellular phones and mobile computers that are part of the ubiquitous computing paradigm can be classified as low power devices due mainly to mechanical constraints. In cryptographic research, many schemes exist for speeding up the computational process of generating a digital signature (signing) as well as signature verification. We list several such schemes below:

The RSA signature scheme with a small public exponent The security of the RSA signature scheme is based on the intractability of the integer factorization problem. If a small number such as 3 or $2^{16}+1$ is chosen as the public exponent of the scheme, then the signature verification scheme is significantly speeded up without any loss of security.

The DSA scheme with precomputation The digital signature algorithm (DSA) is a variant of the ElGamal signature scheme. The security of the ElGamal signature scheme is based on the intractability of the discrete logarithm problem. This scheme permits precomputation of exponentiation for signing leaving only minor calculations in modular arithmetic to be performed on-line.

The ElGamal signature scheme using elliptic curves The discrete logarithm problem which is the basis for ElGamal signature scheme is normally defined over a multiplicative group of a finite field. Koblitz [6] and Miller have described implementation of the ElGamal signature scheme based on elliptic curves over a finite field. It is believed that the discrete logarithm problem for such an elliptic curve group is harder than for a multiplicative group of a finite field This observation allows implementations of ElGamal signature scheme to use a smaller security parameter when using elliptic curves (say 160 bits) than when using multiplicative groups (say 512 bits) for an equivalent strength in security. This reduction in bit length leads to a corresponding increase in computational efficiency for the elliptic curve based scheme.

On-line/off-line digital signatures This scheme introduced by Even, Goldreich and Micali [4] consists of an off-line portion in which a set of validation parameters for a one-time signature scheme is generated. These values are then hashed and the hash value is signed using a public key signature scheme. Although this signature generation part of the scheme is very slow, it is independent of the messages to be signed and hence pre-computed values can be stored for on-line use. The on-line portion of the scheme use the pre-computed validation parameters to generate one-time signatures which is a very fast computation.

Signcryption This scheme introduced by Zheng [11] combines the two step process of digitally signing and then encrypting the signed message into a single cryptographic primitive termed digital signcryption. The cost of signcryption (both in terms of cryptographic computation and bandwidth) is significantly smaller than the addition of costs for signing and encryption.

The above discussed schemes are one approach to supporting computationally intensive cryptographic processing on low power computing devices by improving computational efficiency. An alternative approach to the same problem is to off-load as much of the cryptographic computations as possible from an original signer (or verifier) to a trusted proxy agent.

2.2 Overview of Proxy Delegation Schemes

A proxy signature scheme is only one of many solutions to the general problem of delegation. Informally stated, the delegation problem is for a verifier to be able to establish the truth or falsity of a claim made by a proxy agent. In the case of digital signatures, a solution for the delegation problem must allow a proxy agent to verify a proxy signature as originating from the claimed principal and also for a verifier to authenticate the signature on a message. We summarize a few of the major delegation schemes below:

Full delegation In this scheme, Alice sends her secret key (of the public key, secret key pair) to Bob over a secure channel. With full delegation, the signatures made by the principal and the proxy agent are identical. Proxy signing by full delegation is not a desirable scheme as it goes against the basic assumption of cryptographic schemes whereby the secrecy of secret key material should not be compromised. Also, the indistinguishability of an original signature from a proxy signature makes determination of actual signer identity impossible in the event of a dispute.

Partial delegation by proxy signature In [7] Mambo, Usuda and Okamoto presented the proxy signature mechanism in which a principal signature holder can have a trusted proxy agent sign messages on its behalf while satisfying the following conditions:

1. Only the principal or its designated proxy agent can create a valid proxy signature and sign messages. This is the *unforgeability* property for a proxy digital signature.
2. The receiver of a signature can verify that it is a valid (proxy) signature for the principal. This is the *verifiability* property for proxy digital signature.

Proxy signature schemes are different from other related schemes, such as *multisignatures* [1, 3] and *group signatures* [2], which also involve more than one signing party for a signature on a given message.

Proxy signature schemes are useful for secure communication by computing devices lacking the necessary computational power to perform cryptographic computations on an on-line real-time basis. These devices can use a more powerful trusted proxy server to perform required cryptographic computations on their behalf while maintaining certain checks and balances against misuse or abuse of the trust placed on the proxy agent.

In Mambo's proxy signature scheme, Alice computes a new secret σ using her secret key x_a. Alice sends this newly generated proxy secret to Bob over a secure channel. Bob uses this proxy secret σ to sign messages on behalf of Alice. If Carol receives a message that is proxy signed for Alice, she uses public key of Alice y_a in a slightly modified manner (from the normal signature scheme) to verify the signature. This is an important property as it allows a signature to be identified as generated by a proxy agent and allows the principal to determine the identity of the proxy signer (the *identifiability* property). However, this leaves the proxy agent unprotected from proxy signatures created by the principal itself. Thus, the *distinguishability* property for proxy signatures that identifies them from original signatures does not lead to a safe *non-repudiability* property applicable to both the principal and the proxy agent. For this reason, the standard proxy signature scheme is known as a proxy-unprotected scheme. A modified scheme that doesn't allow even the original signer to forge proxy signatures is called a proxy-protected scheme.

Delegation by signed token In this scheme, Alice sends a signed token to Bob over a public channel. The token contains the identities of the principal, the proxy agent and other details such as delegated rights and the token lifetime. When Bob signs a message on behalf of Alice, he has to include the token from Alice in this message. The disadvantages in this signed token scheme are the need for a receiver to verify two signatures (Bob's signature on the message and Alice's signature on the token) and effective doubling of the size of the signature on a message due to the need to include the token with every message.

Another well known system based on tokens is the Kerberos authentication scheme [9]. Kerberos uses signed tokens from authentication servers (AS) to authenticate clients to ticket granting servers (TGS) which in turn issue tokens to clients to access servers.

3 An Efficient Proxy Signature Scheme for Secure and Authenticated Message Transmission

The main parameters used in the following signature scheme are p : a large prime number, q : a large prime factor of $p-1$, g : an integer in $[1,\ldots,p-1]$ with order q mod p, $hash$: a one-way hash function such as SHS, or HAVAL, KH : a Key-ed one-way hash function, (E,D) : the encryption and decryption algorithms of a private key cipher, x_a : Secret key of Alice, a randomly chosen integer, y_a : Public key of Alice ($y_a = g^{x_a} \bmod p$), x_b : Secret key of Bob, a randomly chosen integer, y_b : Public key of Bob ($y_b = g^{x_b} \bmod p$) and m : a message.

3.1 Description of the Proxy Signature Scheme by Mambo

We briefly outline the proxy signature scheme for partial delegation introduced in [7] and on which we base our efficient implementation mechanism.

1. Proxy key generation:
   Alice, the original signer, choose a secret random number x from $[1,\ldots,q]$ and compute $K = g^x \bmod p$ and $x_{ap} = x_a + xK \bmod p-1$. The values (p,q,g) are public parameters of the signature scheme, K is a public value and x_{ap} is a shared secret between the principal and the proxy agent.
2. Proxy key delivery:
   Alice sends the newly generated values (x_{ap}, K) to the proxy agent over a secure channel.
3. Proxy key verification:
   The proxy agent accept x_{ap} as a valid proxy right from Alice, if and only if $g^{x_{ap}} = (y_a \cdot K^K) \bmod p$.
4. Signing by proxy agent:
   When the proxy agent signs a message m on behalf of Alice, he uses the original signature scheme with x_{ap} as the secret signing key to compute the signature value $sig_{ap}(m)$. The proxy signature is $(sig_{ap}(m), K)$.
5. Proxy signature verification:
   A receiver uses the same verification operation of the original signature scheme to verify the signature on message m. However, as the public key of Alice, a newly computed value $y_a \cdot K^K \bmod p$ is used in place of the original y_a.

3.2 Description of the Mambo's Proxy Signature Scheme Adapted for Shortened DSS

Current implementations of signcryption cryptographic primitive is based on shortened ElGamal signature schemes. For the purpose of describing our proxy signature implementation, first we briefly outline the shortened DSS.

`Sign:` Shortened Digital Signature Scheme 1 (SDSS1)
x : a secret random number from $[1, \ldots, q-1]$ chosen independently for each signing operation by Alice, $r = hash(g^x \bmod p, m)$, $s = x/(r + x_a) \bmod q$ and $sig_a(m) \equiv (r, s)$: Alice's signature on message m.

`Verify:` Verification for SDSS1
A verifier recover $k = g^x \bmod p$ from signature (r, s), public parameters (g, p) and public key y_a as $k = (y_a \cdot g^r)^s \bmod p$.
Accept m as a valid message from Alice if and only if $hash(k, m) = r$.

Next, we show the adaptation of Mambo's proxy signature scheme to the SDSS version 1 (SDSS1). A detailed description of SDSS1 is given by Zheng in [11].

`Setup:` This action consists of two steps.
(1) Generation of proxy signature using SDSS1
Alice generate *proxy secret* (x_{ap}, K) for delivery to proxy agent:
x : a secret random number from $[1, \ldots, q-1]$ chosen specifically for creating the proxy right, $K = g^x \bmod p$ and $x_{ap} = x_a + xK \bmod p - 1$. When generating K for granting of proxy rights, there is no specific message involved. In this instance the signature is simply a *signature on original signer's identity*. Now, Alice sends the values (x_{ap}, K) through a secure channel to her proxy agent.
(2) Verification by proxy agent
The proxy agent recovers $k_p = g^{x_{ap}} \bmod p$ from received value K, public parameter p and public key y_a as $k_p = (y_a \cdot K^K) \bmod p$.
Accept x_{ap} as a valid proxy right from Alice if and only if $k_p = g^{x_{ap}}$.

The proxy agent is unable to derive x_a from (x_{ap}, K) due to the intractability of the discrete logarithm problem (DLP). Now, when the proxy agent signs a message m on behalf of Alice, he uses the proxy secret key x_{ap}. A receiver of a signed message $(m, sig_{ap}(m), K)$ can verify its origin using the proxy public key y_{ap} where $y_{ap} = y_a \cdot K^K \bmod p$.

`Sign:` Signing using proxy signature for SDSS1
x' : a secret random number from $[1, \ldots, q-1]$ chosen independently for each signing operation by proxy agent, $r' = hash(g^{x'} \bmod p, m)$, $s' = x'/(r' + x_{ap}) \bmod q$ and $sig_{ap}(m) \equiv (r', s', K)$: Proxy signature of Alice on message m.

`Verify:` Verification using proxy signature for SDSS1
A verifier recover $k = g^{x'} \bmod p$ from signature (r', s', K), public parameters (g, p) and public key of Alice y_a as $k = (y_{ap} \cdot g^{r'})^{s'} \bmod p$.
Accept m as a valid message from Alice if and only if $hash(k, m) = r'$.

3.3 Description of the Proposed Proxy-Signcryption Scheme

We omit a separate description of the implementation of signcryption scheme using SDSS1 as this scheme is used without modification for the sign and verify operations by the proxy agent. Therefore, we directly show the implementation of a proxy signature scheme using signcryption for SDSS1.

`Setup:` Alice creates the proxy secret (x_{ap}, K) and securely transmits it to the trusted proxy agent (we omit the verification of the received secret by the proxy agent).
x : a secret random number from $[1, \ldots, q-1]$ chosen specifically for creating the proxy right, $K = g^x \bmod p$ and $x_{ap} = x_a + xK \bmod p-1$. The proxy public key $y_{ap} = y_a \cdot K^K \bmod p$.

`Sign:` Signcryption by Alice's proxy using the proxy signature scheme
x' : a secret random number from $[1, \ldots, q-1]$ chosen independently for each signing operation by the proxy agent, $k = y_b^{x'} \bmod p$, $k_1 \parallel k_2 = k$, $r' = KH_{k_2}(m)$ or equivalently $hash(k_2, m)$, $s' = x'/(r' + x_{ap}) \bmod q$, $c = E_{k_1}(m)$ and $cryptogram_{ap}(m) \equiv (c, r', s', K)$: Proxy signed, signcrypted message m from Alice to Bob.

`Verify:` Unsigncryption by Bob using the proxy signature scheme
Recover k from signature (r', s', K), public parameters (g, p), public key of Alice y_a and secret key of Bob x_b as $k = (y_{ap} \cdot g^{r'})^{s' \cdot x_b} \bmod p$.
Split k into k_1 and k_2. Recover the message $m = D_{k_1}(c)$. Accept m as a valid message from Alice if and only if $KH_{k_2}(m) = r'$.

4 Performance of the Proxy-Signcryption Scheme

Use of signcryption gives both computational and storage cost savings to the proposed proxy signature scheme over the original scheme used to derive it. For the purpose of performance analysis we briefly outline a protocol model for message transfer through trusted proxy agents by two end-point principals.

4.1 The Proxy Agent based Communication Protocol Model

The simple communication protocol described below assumes that the principal and the proxy agent have already established a secure channel for message transmission between them. This secure channel could be constructed by exchanging a shared secret sk as the session key by using public key cryptography. Once a secure channel is available, before proxy signed message transmission could begin, both the principal and the proxy agent perform the `Setup` step to generate, transfer and verify the proxy secret key. The numbered steps in figure 1 refer to only one side of the communication protocol which is symmetric for sender and receiver end-points.

1. Alice uses the shared session key sk and symmetric encryption to transmit the ciphertext $C = (ENC_{sk}(m), KH_{sk}(m))$ to the proxy agent over a secure channel.
2. The proxy agent decrypts C to recover the message and verify its integrity using the key-ed hash function. Then it uses the `Sign` step to create the proxy-signcrypted ciphertext $cryptogram_{ap}(m)$.

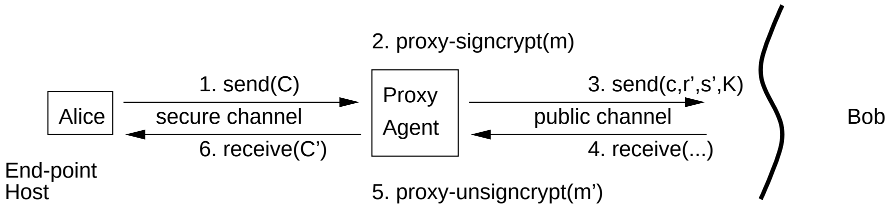

Fig. 1. Proxy-signcrypted message transfer protocol

3. The cryptogram is transmitted to its destination over a public channel.
4. In the reverse process, the proxy agent receives a proxy-signcrypted, signcrypted or normally signed and encrypted message m' addressed to its end-point principal.
5. Assuming a proxy-signcrypted message, the proxy agent uses the `Verify` step to recover and authenticate the message. If message recovery action is successful, then the proxy agent uses sk and symmetric encryption to prepare the message for transmission onwards to Alice. If the authentication fails, it may initiate an alternative action such as retry.
6. Alice receives the incoming message from the proxy agent over the secure channel and decrypts it to recover the plain text message m'.

4.2 Computational Efficiency of the Proxy-Signcryption Scheme

There is no extra cost, over the traditional signing costs, incurred by a proxy agent for repeated signing of messages using proxy-signcryption as the algorithm used by a proxy agent for signing is same as that for an original signer. The principal on whose behalf the proxy agent signs has a one-time cost in generating the proxy secret (x_{ap}, K). The proxy agent has a one-time cost in verifying the proxy secret and generating the proxy public key value y_{ap} (assuming that we let the more powerful proxy server compute the public key value. Alternatively, a receiver can compute the modified proxy public key value itself using the received K value and the original sender's public key y_a). These three operations, which are identical to Mambo's scheme, form the fixed cost component of the proxy-signcryption scheme.

For concreteness, assume following values for the security parameters: $|p| = 768$, $|q| = 152$ and $|KH(\cdot)| = |hash(\cdot)| = 80$. All the values are in bits and consistent with the minimum values recommended for security parameters for current practice.

The `Setup` step in which Alice generates the proxy secret requires one modular exponentiation, one 152-bit modular multiplication and one addition. The proxy secret verification requires an additional two modular exponentiations and one 152-bit modular multiplication. Furthermore, generation of the proxy-public key requires one modular exponentiation and one 152-bit modular multiplication. Considering one modular exponentiation to be equivalent to 240 modular multiplications on average [8, page 453], when using standard exponentiation techniques, the total fixed cost is 963 modular multiplications.

The proxy-signcryption step for Alice requires one modular exponentiation, one 152-bit modular inversion, one addition, one hash computation with $|m|$ bit long input and one symmetric encryption of a $|m|$ bit long input. However, when using a standard proxy signature scheme such as [7], only the signature is computed. If DSS is used as the underlying signature scheme, the signature generation requires one modular exponentiation, one 152-bit modular inversion, two 152-bit modular multiplications, one addition and one hash computation with $|m|$ bit long input. Thus, for nearly the cost of only proxy signing with respect to standard schemes, proxy-signcryption achieves both signature and encryption. This is approximately a 50% saving in computational cost for transmission of a secure and authenticated message.

A similar improvement in computational cost for proxy-unsigncryption operation, in comparison to standard proxy signature verification schemes, is gained due to the computational symmetry of the protocol.

4.3 Transmission Efficiency of the Proxy-Signcryption Scheme

The length of a proxy signature expands by $|p|$ bits over the underlying ordinary signature scheme as a result of the inclusion of K value in the transmitted signature. This is true for both Mambo's original proxy signature scheme and the proxy-signcryption scheme described earlier in section 3.3. However, in practical proxy signature schemes a verifier may opt not to calculate the proxy public key itself using the received K value but retrieve a pre-computed proxy public key from a certification authority (CA). A main reason for this would be to non-interactively determine if the delegated proxy rights have been revoked by the original signer. In this case the transmission of the K value can be omitted from the proxy signature thus compressing it to the cryptogram (c, r', s'). If the protocol infrastructure is constructed in this manner, proxy-signcryption achieves the same degree of bandwidth savings as the underlying signcryption primitive when compared against, for example, Mambo's proxy signing followed by ElGamal encryption.

5 Security of the Proxy-Signcryption Scheme

In general, digital signature schemes based on public key cryptography make trust assumptions only for the two end-points of an interaction. This two-party trust model assumes the end-points to be able to secure their signing keys by not compromising their secrecy. The trust model can be extended to include third-party trusted certificate authorities in which public keys are held formatted as digital certificates for secure distribution. In proxy signature schemes, the general two-party trust model is expanded to a three-party model that includes a proxy agent. Each principal has to trust its proxy agent to both secure the proxy secret and to function honestly. In practice, this expanded trust model causes the principal to make a trade-off between security and speed of operation. Interestingly, the class of end-point hosts that we focus on this paper, low power devices, are inherently harder to secure against concerted attacks. In this scenario, the trust trade-off allows security related operations to be performed primarily at a more powerful proxy agent which can be better protected than a weaker end-point principal.

The strength of the proxy signature scheme on which we based our implementation is not greater than the strength of the underlying ordinary signature scheme. For the proxy-signcryption scheme described earlier, the *cryptographic strength* is same as for the general signcryption primitive which in turn is based on the intractability of discrete logarithm problem. As explained previously in section 2.2, the proxy signature scheme has several properties that make it resistant to a range of attacks. Following discussion informally shows that these properties prevent the weakening of the underlying signature scheme when implemented as a proxy signature scheme by safeguarding against both outsider and insider attacks.

1. As Alice's secret x_a can not be computed from the proxy secret (x_{ap}, K) and publicly known parameters, a corrupt proxy agent can not forge delegation of signature rights by itself. This prevents both illegal acquisition and propagation of proxy rights by chaining or nesting [10]. Also, a malicious third party that breaks-in to the principal to steal the secret x_a can not duplicate a previously generated proxy secret without knowing the random value x used in the key generation. Therefore a proxy-signcryption by a proxy agent is *unforgeable* except in the case of a malicious principal.
2. As the signature verification is distinctly different for an original signature and a proxy signature, a corrupt proxy agent can not deny authorship of a signature without claiming complete loss of its proxy secret. In combination with these *distinguishability* and *identifiability* properties, proxy-signcryption in a sparse message-space (i.e. message text can be meaningfully interpreted) gives a strong *undeniability* property to the proxy agent generated signatures.

Finally, an important issue of proxy signatures is the revocation of a proxy right granted to a proxy agent. A straightforward mechanism is to revoke the principal's public key, thus preventing future acceptance of proxy signatures by recipients as they can no longer be correctly verified. However, we consider this revocation mechanism to be unsuitable as a more meaningful action would be for an original signer to maintain a long-lived public-private key pair and periodically generate proxy rights with shorter lifetimes. Another approach is to limit the duration of the proxy right delegation by specifying a validity period in a digital certificate used for the distribution of proxy-public key and have a verifier obtain the proxy public key from a CA.

6 Conclusion

We have presented an implementation of a new proxy sign-and-encrypt scheme for secure message transmission by combining the proxy signature and signcryption public key cryptography paradigms. The proposed scheme is shown to be efficient in terms of computation and communication costs. The security of the proposed scheme was discussed informally and we consider a formal proof of security to be important future work. In the full version of this paper [5], we have illustrated a possible use for the proposed scheme in a GSM application scenario that uses short messages which are individually secured. Many applications from the area of electronic-commerce that can be implemented for a mobile clientele require large number of individual short messages for the completion of a transaction. These include banking services, stock trading, international roaming information for GSM, etc. Apart from the low power capabilities of the mobile devices that make the use of proxy signature generation useful, many of the messages used in these applications carry information that requires to be kept confidential. Thus, the additional capability of content encryption provided by proxy-signcryption at no extra cost (compared to normal proxy signature schemes) is a useful feature. The scheme is specially suited for secure application development for a network communication model based on an unreliable datagram transfer protocol where a reliable session connection for the entire duration of a transaction is unlikely to be available.

References

[1] C. A. Boyd. Multisignatures revisited. In *Cryptography and Coding III*, pages 21–30, 1993.

[2] D. Chaum and E. van Heyst. Group signatures. In *Advances in Cryptology - EUROCRYPT'91*, volume 547 of *LNCS*, pages 257–265, 1991.

[3] R. A. Croft and S. P. Harris. Public-key cryptography and re-usable shared secrets. In *Cryptography and Coding*, pages 189–201, 1989.

[4] S. Even, O. Goldreich, and S. Micali. On-line/off-line digital signature. *Journal of Cryptology*, 9:35–67, 1996.

[5] C. Gamage, J. Leiwo, and Y. Zheng. An efficient scheme for secure message transmission using proxy-signcryption. Technical Report 98-01, Peninsula School of Computing & Information Technology, Monash University, July 1998.

[6] N. Koblitz. Elliptic curve cryptosystems. *Mathematics of Computation*, 48:203–209, 1987.

[7] M. Mambo, K. Usuda, and E. Okamoto. Proxy signatures: Delegation of the power to sign messages. *IEICE Trans. on Fundamentals*, E79-A(9):1338–1354, 1996.

[8] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone. *Handbook of Applied Cryptography.* CRC Press, 1996.

[9] B. C. Neuman and T. Ts'o. Kerberos: An authentication service for computer networks. *IEEE Communications Magazine*, 32(9):33–38, 1994.

[10] V. Varadharajan, P. Allen, and S. Black. An analysis of the proxy problem in distributed systems. In *Proceedings of the 1991 IEEE Computer Society Symposium on Research in Security and Privacy*, pages 255–275, 1991.

[11] Y. Zheng. Digital signcryption or how to achieve cost(signature & encryption) $\ll$ cost(signature) + cost(encryption). In *Advances in Cryptology - CRYPTO'97*, volume 1294 of *LNCS*, pages 165–179, 1997.