Multipoint-to-Multipoint Secure-Messaging with Threshold-Regulated Authorisation and Sabotage Detection

Alwyn Goh[1] and David C.L. Ngo[2]

[1] Corentix Laboratories, B-19-02 Cameron Towers, Jln 5/58B,
46000 Petaling Jaya, Malaysia.
alwyn_goh@yahoo.co.uk

[2] Faculty of Information Science & Technology, Multimedia University,
75450 Melaka, Malaysia

Abstract. This paper presents multi-user protocol-extensions for Schnorr/Nyberg-Ruepple (NR) signatures and Zheng *signcryption*, both of which are elliptic curve (EC)/discrete logarithmic (DL) formulations. Our extension methodology is based on k-of-n threshold cryptography—with Shamir polynomial parameterisation and Feldman-Pedersen verification—resulting in multi-sender Schnorr-NR (SNR) and multi-sender/receiver Zheng-NR (ZNR) protocols, all of which are interoperable with their single-user base formulations. The ZNR protocol-extensions are compared with the earlier Takaragi et al multi-user *sign-encryption*, which is extended from a base-protocol with two random key-pairs following the usual specification of one each of signing and encryption. Both single and double-pair formulations are analysed from the viewpoint of EC equivalence (EQ) establishment, which is required for rigorous multi-sender functionality. We outline a rectification to the original Takaragi et al formulation; thereby enabling parameter-share verification, but at significantly increased overheads. This enables comprehensive equivalent-functionality comparisons with the various multi-user ZNR protocol-extensions. The single-pair ZNR approach is shown to be significantly more efficient, in some cases demonstrating a two/three-fold advantage.

1 Introduction

The emergence of various technologies ie peer-to-peer computing and *ad hoc* communications motivates the development of transactional models beyond the presently dominant presumption of single-user functionality and point-to-point connectivity. This in turn motivates the development of cryptographic protocols to support network-mediated collaboration and workgroup transactions, the multi-user nature of which is not accommodated naturally by the conventional presumption of user-specific key-parameterisation. External transaction-to-workgroup association is a far better solution—from the viewpoint of transactional logic and liability—which also reduces the receiver-side storage overhead to a single public-key.

The cryptographic specification is therefore to rigorously associate multiple user-specific *key-shares* with a common workgroup public-key, so that a configurable user-subset is able to exercise workgroup-representative authority. This can be ele-

A. Lioy and D. Mazzocchi (Eds.): CMS 2003, LNCS 2828, pp. 27–39, 2003.

gantly implemented via the polynomial-based k-of-n threshold methodology of Shamir [1] and Feldman-Pedersen [2-4], which is applicable to EC [5, 6]/DL protocols. k-of-n thresholding is therefore a useful multi-user specification methodology; as respectively demonstrated by Park-Kurosawa [7] and Takaragi et al [8] extensions on ElGamal [9] and NR [10] signatures respectively, the latter of which was presented to the Institute of Electrical and Electronics Engineers (IEEE) study-group for public-key cryptography standards. Takaragi et al also specifies a sign-encryption protocol able to incorporate multiple senders *and* receivers. This paper departs from earlier work in its emphasis on secure-messaging rather than signatures; with its focus on integration of message authentication/encryption and multi-user functionality.

We outline k-of-n threshold extensions for Schnorr [11], Zheng [12] and NR constructions, with the characteristic property of message-level parameterisation based on a single EC/DL key-pair of the initial sender-determined randomisation. This approach was motivated by the use of the single-pair in Zheng signcryption for both authentication and encryption; which is a departure from the more frequently encountered specification of distinct key-pairs for each message-related functionality, as exemplified by the Takaragi et al NR-derived (TNR) sign-encryption. Single (rather than double) key-pair secure-messaging is significantly more compute-efficient on a point-to-point basis, and is shown in this paper to be similarly advantageous for multi-user extensions. This applies to both *fast* and rigorous multi-sender modes, the latter of which necessitates detection of malformed parameter-shares via ECEQ establishment. The original multi-sender TNR sign-encryption is, in fact, not rigorous due to non-establishment of ECEQ, which can be rectified via application of the Chaum-Pedersen [13] and Fiat-Shamir[14] protocols.

2 Review of Base Protocols and Mechanisms

2.1 Schnorr-Zheng, NR and Takaragi et al Cryptography

All signature and secure-messaging protocols in this section presume prior specification of a EC/DL finite-field. We adopt the former description, denoted F with base-point $\mathbf{g} \in$ F and multiplicative-group $G = \{k\,\mathbf{g} : k \in Z_q\} \subset F$. Schnorr signatures are inherently bandwidth-efficient, with signature bit-length of $|h| + |q|$ (for h some cryptographic hash) independent of the underlying finite-field. Zheng secure-messaging extends the Schnorr formulation to enable receiver-designation, so that the sender-side signcryption and receiver-side unsigncryption operations respectively incorporate symmetric cipher operations ⟨E, D⟩. Both protocols require prior specification of sender (A) key-pair ⟨a, **A** (= a **g**)⟩, with Zheng additionally necessitating receiver (B) key-pair ⟨b, **B** (= b **g**)⟩. Sender and receiver-side computations then proceed as follows: with F some key-formatting function, most conveniently implemented with hash h. Note the use of basepoint **g** and receiver public-key **B** as the expansion point for initial randomisation k, resulting in random message-specific public-keys **k** and **β**. This prescription is entirely consistent with NR cryptography, with the only difference being specification of $r = \nu - h(m)$ instead of the above-outlined $r = h_\nu(m)$.

Table 1. (a) Schnorr and (b) Zheng protocols

	Schnorr	*Zheng*
A	Generate ⟨k, **k** (= k **g**)⟩ Compute ν = F(**k**) Compute $r = h_\nu(m)$ Compute s = k – ar (mod q)	Generate ⟨k, β (= k **B**)⟩ Compute ⟨μ, ν⟩ = F(β) Compute $c = E_\mu(m)$ Compute $r = h_\nu(m)$ Compute s = k – ar (mod q)
	↓ ⟨m, r, s⟩	↓ ⟨c, r, s⟩
B	Recover **k** = s**g** + r**A** Recover ν = F(k) Confirm $h_\nu(m) = r$	Recover β = b (s**g** + r**A**) Recover ⟨μ, ν⟩ = F(β) Recover $m = D_\mu(c)$ Confirm $h_\nu(m) = r$

The computation-overheads of SNR and ZNR are essentially equal from the viewpoint of EC scalar-multiplication (M) operations, each of which is far more expensive than EC point-addition (A) or number-field/symmetric computations. Leading-order analysis then yields sender and receiver-side overheads of M and 2M. ZNR is therefore significantly more compute-efficient compared to the usual superposition of signing and encryption operations. S/ZNR is also more efficient than ElGamal and the USA National Institute of Standards and Technologies (NIST) Digital Signature Standard (DSS), both of which require sender/receiver-side number-field multiplicative inversion.

Both **k** and β have different functional roles, the latter of which enforces receiver-side demonstration of private-key b as a precondition for message-recovery and verification. This is beyond the scope of pure *multisignature* formulations, but is important for collaborative protocols with receiver-designation. The Takaragi et al NR-extended (TNR) *sign-encryption*—with explicit use of **k** for authentication and β for encryption—takes an alternative approach, as outlined below:

Table 2. TNR sign-encryption

A	Generate ⟨k, **k**, β⟩ Compute ν = F(**k**) and μ = F(β) Compute r = ν – h(m) Compute s = k – ar (mod q) Compute $c = E_\mu(m)$
	↓ ⟨c, r, s⟩
B	Recover **k** = s**g** + r**A** and ν Recover β = b **k** and μ Recover $m = D_\mu(c)$ Confirm ν = r + h(m)

This formulation costs 2M on the sender-side and 3M on the receiver-side, the latter of which arises from the necessity to sequentially compute **k** and then **β**. Both are more significantly more compute-intensive than the corresponding ZNR operations.

2.2 k-of-n Polynomial Thresholding

k-of-n threshold cryptography as formulated by Shamir allows workgroup (set of all users U) key-parameterisation via (k–1)-degree polynomial $e(x) = \sum_{\mu=0}^{k-1} e_\mu x^\mu \pmod q$, with a = e(0) mod q interpreted as the workgroup private-key. Individual users—of which there are n, indexed i ∈ U—would then be assigned polynomial-associated private key-shares $a_i = e(i) \bmod q$, which are essentially a k-th share of a if e is secret. This arises from the necessity of at least k datapoints of form ⟨i, e(i)⟩ for finite-field Lagrange interpolation ie $e(x) = \sum_{i \in S} e(i) \left(\prod_{j \in S-\{i\}} \frac{x-j}{i-j} \right) \pmod q$. Evaluation of this expression results in $e(0) = a = \sum_{i \in S} \varepsilon_i a_i \pmod q$ with index-coefficient $\varepsilon_i = \prod_{j \in S-\{i\}} \frac{j}{j-i} \pmod q$ for any k-sized subset S ⊂ U. Knowledge of e should be restricted to a trusted key-generator (T), whose role will be subsequently outlined.

Pedersen verification allows individual key-shares a_i to be verified as a k-th portion of workgroup private-key a without divulging polynomial e. This operation can be executed with [3] or without [4] a centralised T. Presumption of T allows an efficient non-interactive implementation; with individual EC key-pairs $\langle a_i, \mathbf{A}_i (= a_i \mathbf{g}) \rangle$ and polynomial parameterisation $\langle e_\mu, \mathbf{e}_\mu (= e_\mu \mathbf{g}) \rangle$, the latter of which includes workgroup key-pair ⟨a, **A** (= a **g**)⟩. Key-share generation, distribution and verification between T and all users i ∈ U then proceeds as follows:-

Table 3. Key-share generation, distribution and verification

T	Generate polynomial $\langle e_\mu, \mathbf{e}_\mu \rangle$ Generate key-share $\langle a_i, \mathbf{A}_i \rangle$ for $\forall i$ Authenticated ch: $\Downarrow \mathbf{e}_\mu, \langle i, \mathbf{A}_i \rangle$	Secure ch: $\downarrow a_i$
i ∈ U	Confirm $\sum_{\mu=0}^{k-1} (i^\mu \bmod q) \mathbf{e}_\mu = a_i \mathbf{g} = \mathbf{A}_i$	

the last step of which is a zero knowledge (ZK) verification of key-share possession by user i, thereby enabling engagement in the subsequently outlined protocols. Note the non-interactive nature of the above-described one-time procedure, with authenticated communication essentially equivalent to signed postings on a bulletin-board.

3 Basic Multi-sender Protocol-Extensions

3.1 Individual and Workgroup Parameterisations

The most straightforward extension methodology would be via SNR **k** and ZNR β public-keys as the starting point. The protocol parameters are outlined below:

Table 4. Sender-specific and workgroup-combined parameters

	SNR	*ZNR*
i∈ S	k_i	
	$\mathbf{k}_i = k_i \mathbf{g}$	$\beta_i = k_i \mathbf{B}$
S ⊂ U	$\mathbf{k} = \sum_{i \in S} \mathbf{k}_i$	$\beta = \sum_{i \in S} \beta_i$
	ν, r	$\langle \mu, \nu \rangle, \langle c, r \rangle$
i	$s_i = k_i - \varepsilon_i a_i r \pmod q$	
S	$s = \sum_{i \in S} s_i \pmod q$	

with Schnorr-Zheng/NR differentiation via specification for r. The end result would be SNR signature ⟨r, s⟩ or ZNR signcryption ⟨c, r, s⟩, as would be computed by an entity with private-key $a = \sum_{i \in S} \varepsilon_i a_i \pmod q$. This approach has been demonstrated by Takaragi et al to be advantageous compared to the earlier Park-Kurosawa formulation with individual random polynomials.

The Takaragi et al description of multisignature formation specifies broadcast of all individual $\langle \mathbf{k}_i, s_i \rangle$ and repeated computation of the common **k** and ⟨r, s⟩ by each i ∈ S. We outline an alternative presentation with a centralised combiner (C) of workgroup parameters—the details of which can logged and straightforwardly verified—which is also applicable towards TNR multisignatures, as demonstrated below: (See Table 5) resulting in NR signature ⟨r, s⟩. Such an implementation clearly and efficiently separates security-critical sender-specific and verifiable workgroup-aggregated operations, the latter of which does not result in an externally (to the workgroup) visible contribution.

Table 5. TNR multisignature formation

	$i \in S$		C
1	Generate $\langle k_i, \mathbf{k}_i \rangle$	$\mathbf{k}_i \rightarrow$	
2		$\Leftarrow \langle \forall i, r \rangle$	Compute k, ν and r
3	Compute ε_i and s_i	$\mathbf{S}_i \rightarrow$	Compute $\forall \varepsilon_i$
4			Confirm $s_i \mathbf{g} + r\varepsilon_i \mathbf{A}_i = \mathbf{k}_i$ Compute s

3.2 Multi-sender Extended Cryptography

Recall that TNR multisignatures are an extension of the NR base-formulation, hence the applicability of Table 5 to Schnorr multisignatures via definition $r = h_\nu(m)$. The equivalent ZNR extension is as follows:-

Table 6. ZNR multi-signcryption

	$i \in S$		C
1	Generate $\langle k_i, \beta_i \rangle$	$\beta_i \rightarrow$	
2		$\Leftarrow \langle \forall i, r \rangle$	Compute β, ⟨μ, ν⟩ and ⟨c, r⟩
3	Compute ε_i and s_i	$s_i \rightarrow$	Compute $\forall \varepsilon_i$
4			Compute s

Both T/SNR and ZNR formulations have sender-side overheads of M (computations) and $|p| + |q|$ (communications), which is slightly higher (with respect bandwidth) compared with the single-sender base-protocols in Table 1. The 2kM computation required for T/SNR signature-share verification in step (4) of Tables 5 is noteworthy, as is the modest $\left(\frac{k}{2}+1\right)|h|$ broadcast overhead after step (2) in both protocol-extensions.

Note that submission of **k** after step (1) and its subsequent verification in step (4) as in Table 5, does not preclude protocol-sabotage by individual users. This is executed via submission of $\beta = k\,\mathbf{B}$ and $s' = k' - \varepsilon ar \pmod q$ with different initial randomisations, resulting in receiver-side inability to recover the signcrypted message. Detection and mitigation of malformed parameter-shares motivates our subsequent analysis of TNR sign-encryption, and formulation of a ZNR extension with verified combination.

4 Multi-sender Protocol-Extension with Verified Combination

4.1 Analysis of Randomised Key-Pairs

Verification of the ZNR-shares in Table 6 essentially requires establishment that the public-keys $\langle \mathbf{k}, \beta \rangle$ are ECEQ . This is not demonstrated in TNR multi-sender sign-encryption—which simply uses one key-pair each for parameter-share authentication and encryption—as outlined below:-

Table 7. TNR multi-sender sign-encryption

	$i \in S$		C
1	Generate $\langle k_i, \mathbf{k}_i, \beta_i \rangle$	$\langle \mathbf{k}_i, \beta_i \rangle \rightarrow$	
2		$\Leftarrow \langle \forall i, r \rangle$	Compute $\mathbf{k}$, ν and r
3	Compute ε_i and s_i	$s_i \rightarrow$	Compute $\forall \varepsilon_i$
4			Confirm $s_i \mathbf{g} + r\varepsilon_i \mathbf{A}_i = \mathbf{k}_i$ Compute s Compute β, μ and c

Note the pair-related computations are essentially independent signing and encryption operations—with increased sender-side overheads of 2M and $2|p| + |q|$—which is problematic due to individual senders being able to sabotage the protocol through submission of non-ECEQ pair $\langle \mathbf{k}, \beta' \rangle$. Such an malformed submission enables successful verification (internal to the workgroup), but prevents proper receiver-side recovery (typically outside the workgroup). Saboteurs can therefore remain undetected in TNR multi-sender sign-encryption.

This inability to detect non-ECEQ pairs prior to combination is unfortunate, since typical operations might result in submission of more than k parameter-shares. Combiner-side detection of sabotaged parameter-shares under such circumstances would therefore allow for their straightforward replacement with well-formed ones, so that the resultant $\langle c, r, s \rangle$ is also well-formed. Lack of such a capability, on the other hand, is problematic in any number of realistic operational scenarios.

4.2 Rectification via ECEQ Establishment

A pair P = $\langle \mathbf{k}, \beta \rangle$ can be proven ECEQ with respect basepoint pair $\langle \mathbf{g}, \mathbf{B} \rangle$ via the Chaum-Pedersen [13] protocol, which can be made non-interactive via Fiat-Shamir [14] heuristics. Prover (P) knowledge of common randomisation k allows Verifier (V) side confirmation of ZK proof $\langle e, z \rangle$ as follows:-

Table 8. ECEQ of P with respect $\langle \mathbf{g}, \mathbf{B} \rangle$

P	Generate random r Compute $P' = \langle r\,\mathbf{g}, r\,\mathbf{B} \rangle$ Compute $e = h(\mathbf{g}, \mathbf{B}, P, P')$ Compute $z = r - ek \pmod q$
	$\downarrow \langle e, z \rangle$
V	Compute $\mathbf{k}' = e\mathbf{k} + z\mathbf{g}$ Compute $\beta' = e\beta + z\mathbf{B}$ Confirm $e = h(\mathbf{g}, \mathbf{B}, P, P')$

which requires prover and verifier-side computation overheads of 2M and 4M respectively, in addition to bandwidth $|h| + |q|$. ECEQ establishment allows rectification of the TNR formulation in Table 7 as follows:-

Table 9. TNR multi-sender sign-encryption with ECEQ

	$i \in S$		*C*
1	Generate $\langle k_i, \mathbf{k}_i, \beta_i \rangle$	$\langle \mathbf{k}_i, \beta_i \rangle \rightarrow$	
2		$\Leftarrow \langle \forall i, r \rangle$	Compute $\mathbf{k}$, ν and r
3	Compute $\langle e_i, z_i \rangle$		Compute $\forall \varepsilon_i$
	Compute ε_i and s_i	$\langle e_i, z_i, s_i \rangle \rightarrow$	
4			Establish ECEQ $\langle \mathbf{k}_i, \beta_i \rangle$ Confirm $s_i\mathbf{g} + r\varepsilon_i \mathbf{A}_i = \mathbf{k}_i$ Compute s Compute β, μ and c

resulting in a well-formed $\langle c, r, s \rangle$; but at significantly higher overheads, particularly combiner-side for large k.

4.3 Homomorphic ECEQ Establishment

ECEQ establishment for multi-sender signcryption is far more straightforward via reexpression of the EC verification condition (V): $s\mathbf{g} + r\mathbf{A} = \mathbf{k}$ (ref Table 1), specifically its RHS(V): $\mathbf{k} = \beta + \delta$ with $\delta = k\,\mathbf{d}$ and $\mathbf{d} = \mathbf{g} - \mathbf{B}$. Individual senders would therefore need to compute and transmit ECEQ pair $\langle \beta, \delta \rangle$, the latter of which essentially constitutes a homomorphic commitment on the former. This results in the following ZNR extension:

Table 10. ZNR multi-signcryption with verified combination

	$i \in S$		C
1	Generate $\langle k_i, \beta_i, \delta_i \rangle$	$\langle \beta_i, \delta_i \rangle \rightarrow$	
2		$\Leftarrow \langle \forall i, r \rangle$	Compute β, $\langle \mu, \nu \rangle$ and $\langle c, r \rangle$
3	Compute ε_i and s_i		Compute $\forall \varepsilon_i$
		$s_i \rightarrow$	Recover $\mathbf{k}_i = \beta_i + \delta_i$
4			Confirm $s_i \mathbf{g} + r\varepsilon_i \mathbf{A}_i = \mathbf{k}_i$
			Compute s

with parameter-share verification in step(4) prior to computation of the workgroup s. The single key-pair computation results in sender-side overheads essentially equal to weak TNR sign-encryption *without* ECEQ (Table 7), but is only half that of the rigorous variant with ECEQ (Table 9). The combiner-side overhead is essentially equal to that of the T/SNR multisignature scheme in Table 5, and also only a third of TNR sign-encryption with ECEQ.

Note the differences in the ECEQ establishment mechanisms, with independent use of $\langle \mathbf{k}, \beta \rangle$ resulting in the necessity for specification of another pair $\langle \mathbf{k}', \beta' \rangle$. ZNR predication on single public-key β, on the other hand, allows for a much simpler homomorphic establishment of ECEQ which leverages EC verification (in any case required) of individually submitted s. This illustrates the efficacy of the ZNR signcryption approach which integrates signature and encryption operations.

5 Multi-receiver Protocol-Extension

5.1 Individual and Workgroup Parameterisations

ZNR multi-receiver extensibility is predicated on receiver-specific ($i \in R$) knowledge of key-share b_i applied to compute parameter-share $\beta_i = b_i(s\mathbf{g} + r\mathbf{A})$. Sufficient quantities of the latter can be summed to obtain workgroup-common ($R \subset U$) $\beta = \sum_{i \in R} \varepsilon_i \beta_i$. This parameterisation also applies to the TNR decrypt-verify protocol, but is beyond the functional scope of the TNR and SNR multisignature formulations.

Following the sender-side analysis, we adopt a presentation with centralised C so as to separate security-critical receiver-specific (predicated on key-share knowledge) and verifiable workgroup-aggregated operations. This is straightforward for ZNR recovery of β, but more complicated for the equivalent TNR operation predicated on both $\mathbf{k}$ and β. The most efficient approach is to independantly compute receiver-specific β_i —departing from single-receiver case in Table 2—and workgroup-common **k** = s**g** + r**A** as illustrated below:

Table 11. TNR multi-receiver decrypt-verification

$i \in R$	Compute β_i
	$\downarrow \beta_i$
C	Compute $\forall \varepsilon_i$, β and μ
	Recover $m = D_\mu(c)$
	Compute **k** = s**g** + r**A** and ν
	Confirm $\nu = r + h(m)$

with an overhead of 2M per receiver (ref Section 2.1), and an additional 2M at C. This is less efficient than multi-receiver ZNR, as will be subsequently demonstrated. Successful message recovery/verification presumes proper sender-side formation of $\langle c, r, s \rangle$, which places a premium on parameter-share verification.

5.2 Multi-receiver Extended Cryptography

ZNR unsigncryption as outlined in Section 2.1 can be extended to incorporate multiple receivers, as follows:-

Table 12. ZNR multi-unsigncryption

$i \in R$	Compute β_i
	$\downarrow \beta_i$
C	Compute $\forall \varepsilon_i$ and β
	Recover $\langle \mu, \nu \rangle = F(\beta)$
	Recover $m = D_\mu(c)$
	Use ν to confirm r

with Schnorr-Zheng/NR differentiation only in the final confirmation ie $h_\nu(m) = r$ and $\nu = r + h(m)$ respectively. This formulation can be used in conjunction with single/multi-sender signcryption protocols of Tables 1(b), 6 and 10; the last of which prevents protocol-sabotage via malformed signcryption-shares. This ZNR extension is also more compute-efficient on the combiner-side—by 2M, due to non-computation of **k**—compared with the equivalent TNR operation.

6 Comparison with TNR Protocols

The computation and communications overheads of the featured multi-user extensions are as follows:-

Table 13. Comparison of (a) single/multi-sender signature/signcryption protocols, and (b) single/multi-receiver verification/unsigncryption protocols.
#, * and + denote receiver–designation, parameter–share verification and receiver–confirmation

Protocol	*Table*	*Sender overhead*	*Combiner overhead*
SNR sgn	1(a)	M, $\lvert h\rvert+\lvert q\rvert$	n/a
ZNR sgncpt $^{\#}$	1(b)		
TNR sgn/enc $^{\#}$	2	2M, $\lvert h\rvert+\lvert q\rvert$	
T/SNR multisgn *	5	M, $\lvert p\rvert+\lvert q\rvert$	2kM, $\left(\frac{k}{2}+1\right)\lvert h\rvert$
TNR multi-sgn/enc $^{\#}$	7	2M, 2$\lvert p\rvert+\lvert q\rvert$	
TNR multi-sgn/enc ECEQ $^{\#*}$	9	4M, 2$\lvert p\rvert$+2$\lvert q\rvert+\lvert h\rvert$	6kM, $\left(\frac{k}{2}+1\right)\lvert h\rvert$
ZNR unverif multi-sgncpt $^{\#*}$	6	M, $\lvert p\rvert+\lvert q\rvert$	kA , $\left(\frac{k}{2}+1\right)\lvert h\rvert$
ZNR verif multi-sgncpt $^{\#*}$	10	2M 2$\lvert p\rvert+\lvert q\rvert$	2kM, $\left(\frac{k}{2}+1\right)\lvert h\rvert$

Protocol	*Table*	*Receiver overhead*	*Combiner overhead*	*Receiver-confirmation*
T/SNR verif	1(a)	2M	n/a	no
ZNR unsgncpt $^{+}$	1(b)			yes
TNR dec/verif $^{+}$	2	3M		
TNR multi-dec/verif $^{+}$	11	2M, $\lvert p\rvert$	2M+kA	
ZNR multi-unsgncpt $^{+}$	12		kA	

Note the presentation of *two* ZNR multi-sender extensions, the more rigorous (Table 10) of which facilitates parameter-share verification in addition to receiver-designation. This is achieved efficiently via homomorphic ECEQ, resulting in overheads only marginally greater than T/SNR multisignature formation (Table 5). Rigorous multi-sender TNR (Table 9) sign-encryption requires significantly higher (doubled/tripled) overheads due to the necessity to establish ECEQ of the $\langle \mathbf{k}, \beta\rangle$ public-keys with respect a challenge (r-dependent) pair $\langle \mathbf{k}', \beta'\rangle$. Both ZNR and TNR multi-sender extensions can be operated in unverified modes ie Tables 6 and 7 respectively, with dispensation of the combiner-side overhead for the latter. ZNR multi-signcryption is also significantly more efficient sender-side when operated in *fast* mode.

The multi-receiver ZNR (Table 12) and TNR (Table 11) formulations differ through their respective use of single $\boldsymbol{\beta}$ and double $\langle \mathbf{k}, \boldsymbol{\beta} \rangle$, the former of which is more efficient. Both protocol-extensions are vulnerable to sender-side sabotage resulting in malformed secure-messages, which emphasises the importance of parameter-share verification. Multi-receiver ZNR in conjunction with the verifying multi-sender and single-sender ZNR variants, can therefore be characterised as rigorous and efficient multipoint-to-multipoint secure-messaging.

7 Concluding Remarks

The outlined multi-user S/ZNR protocols are functionally comprehensive, compute/bandwidth-efficient and transparently interoperable with respect their single-user base-formulations. This allows for straightforward implementation of both within typical workgroup environments; with verified combination by designated users or centralised servers, and externally-visible S/ZNR parameters structurally identical to their single-user base-formulations. Combiners can therefore be regarded as workgroup gateways, the efficiency of which is enhanced by the near-similarity of the S/ZNR formulations. Note the receiver-side operation can be concluded after a single cryptographic computation, and is therefore inherently efficient independant of k. Sender-side collaboration can also be simplified to a single pass for the $(k = 2)$ case, with only initiating $(i \in S)$ and responding $(j \in S)$ users.

This versatility and efficiency stems from the featured multi-user extension methodology on single key-pair base-protocols, which in the case of ZNR departs from the usual prescription (adopted for TNR sign-encryption) of distinct pairs for message-authentication and encryption. The proposed formulation integrates authentication and encryption functionalities, and enables efficient detection of sabotaged parameter-shares in multi-sender ZNR. This capacity for sabotage-detection is also present in the T/SNR multisignature protocol, which is a single-pair authentication-only formulation. Sabotage-detection can also be incorporated into the double-pair multi-sender TNR sign-encryption, but only at the cost of significantly higher overheads compared to multi-sender ZNR signcryption. It is interesting to speculate whether other double-pair secure-messaging formulations can be efficiently extended to incorporate this attribute.

Parameter-share verification is a significant functional advantage, the lack of which jeopardises multi-receiver message-recovery/verification. This can be seen from transaction scenarios featuring long-term—so that existence of the original message, sender-side key-shares or even the sending-workgroup cannot be presumed—escrow of inadvertently malformed secure-messages, resulting in permanent information loss. Efficiency with respect sabotage-detection is also important, especially in consideration of the two/three-fold differences in the ZNR and TNR overheads. The presented ZNR extension can therefore be safely characterised as rigorous yet efficient multipoint-to-multipoint secure-messaging.

References

1. A Shamir (1979). *How to Share a Secret*. Assoc Comp Machinery (ACM) Comms vol 22, no 11: pp 612–613
2. P Feldman (1987). *A Practical Scheme for Non-Interactive Verifiable Secret-Sharing*. 28-th IEEE Symp on the Foundations of Comp Sc: pp 427–437
3. TP Pedersen (1991). *Distributed Provers with Applications to Undeniable Signatures*. Eurocrypt-91, Springer-Verlag Lecture Notes in Computer Science (LNCS) 547: pp 221–238
4. TP Pedersen (1991). *A Threshold Cryptosystem without a Trusted Party*. Eurocrypt-91, Springer-Verlag LNCS 547: pp 522–526
5. AJ Menezes (1993). *Elliptic Curve Public-Key Cryptosystems*. Kluwer Acad Press.
6. IF Blake, G Seroussi & NP Smart (1999). *Elliptic Curves in Cryptography*. Cambridge Univ Press
7. C Park & K Kurosawa (1996). *New ElGamal-Type Threshold Digital Signature Scheme*. Inst Electrical, Info & Comms Engineers (IEICE) Trans, Vol E79-A, no 1: pp 86–93
8. K Takaragi, K Miyazaki & M Takahashi (1998). *A Threshold Digital Signature Issuing Scheme without Secret Communication*. Presentation IEEE P1363 Study Group for Public-key Crypto Stds
9. T ElGamal (1985). *A Public-Key Cryptosystem and Signature Scheme Based on Discrete Logarithms*. IEEE Trans Info Theory
10. K Nyberg & R Ruepple (1993). *A New Signature Scheme Based on DSA Giving Message Recovery*. 1-st ACM Conf on Comp & Comms Security, ACM Press: pp 58–61
11. CP Schnorr (1989). *Efficient Identification and Signatures for Smartcards*. Crypto-89, Springer-Verlag LNCS 435: pp 239–252
12. Y Zheng (1997). *Digital Signcryption or how to Achieve Cost(Signature & Encryption) << Cost(Signature) + Cost(Encryption)*. Crypto-97, Springer Verlag LNCS 1396: pp 291–312
13. DL Chaum & TP Pedersen. *Wallet Databases with Observers*. Crypto-92, Springer Verlag LNCS 740: 89–105
14. A Fiat & A Shamir. *How to Prove Yourself: Practical Solutions to Identification and Signature Problems*. Crypto-86, Springer Verlag LNCS 263: 186–194