Fast Multiparty Threshold ECDSA with Fast Trustless Setup

Rosario Gennaro
City College of NY

Steven Goldfeder
Cornell Tech

Digital Signature Algorithm (DSA)

Given

- a group G of order N
- a generator g
- a private key $\boldsymbol{x}$

To sign a message m:

- pick a nonce $\boldsymbol{k}$ s.t. $1 \leq k \leq q - 1$
- $R = g^{\boldsymbol{k}}$
- $s = \boldsymbol{k}^{-1}(m + \boldsymbol{x} \cdot r) \bmod q$

Signature is (r,s)

ECDSA is DSA over an elliptic curve group

GJKR Threshold DSA

Includes multiplication of Shamir shares

R. Gennaro , S. Jarecki, H. Krawczyk and T. Rabin. *Threshold DSS Signatures*. EUROCRYPT ‘96.

Shamir's Secret Sharing (Shamir'79)

- If you have a secret s
  - an integer modulo a prime q
- Consider the polynomial $F(x)=a_0+a_1x+\ldots+a_tx^t$
  - where $a_0=s$
- Give player P_i the share $s_i=F(i)$
  - $t+1$ players can recover the secret
  - t or less have no information about s
    - any value is consistent with their shares

Addition of shares is easy

- If you have two secrets a,b shared via Shamir
  - a, with polynomial $F(x)$ and shares a_i
  - b, with polynomial $G(x)$ and shares b_i
- Players can reconstruct $c=a+b$ by
  - revealing $c_i=a_i+b_i$
  - A point on the polynomial $(F+G)(x)$
  - still of degree t
  - no other information about a,b is released

$r = g^{k}$
$s = k^{-1}(m + \boldsymbol{x} \cdot r) \bmod q$

Problem: Multiplication

If a and b are shared on degree t polynomials

$a \times b$ will be shared on a degree $2t$ polynomial

➔ Need $2t + 1$ players to sign

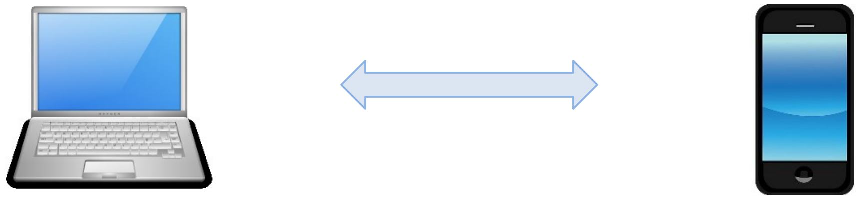
BUT $t + 1$ corrupted players can compromise security!

Requires extra participants

Need $2t + 1$ players to sign

BUT $t + 1$ corrupted players can compromise security

2-out-of-2 threshold not possible



Threshold optimality

Given a (t, n)-threshold signature scheme, obviously t + 1 honest players are necessary to generate signatures. We say that a scheme is ***threshold-optimal*** if t + 1 honest players also suffice.

Previous work

t-out-of-n: GGN16, B**G**G17

However it required a dealer to generate and share the secret key x to the players (in practice)

2-out-of-2: MR01, L17, D+18

Multiplicative-to-additive conversion (MtA)

a

$\boxed{s = a \times b}$

b

c_1 →

← c_2

$b' = \text{func}(c_1,c_2)$

$a' = \text{func}(c_1,c_2)$

$\boxed{a' + b' = a \times b = s}$

Additively Homomorphic Encryption

- An encryption scheme E such that if $c_1 = E(m_1)$ and $c_2=E(m_2)$ then
  - there exists an operation $\oplus$ such that
    - $c_1 \oplus c_2 = E(m_1+m_2 \bmod N)$

- Note that this means that if a is an integer we can also compute
  - $E(am_1) = c_1 \oplus \ldots \oplus c_1 = a \otimes c_1$

- Example: Paillier’s encryption scheme where N is an RSA modulus.

Multiplicative-to-additive conversion (MtA -- Gilboa)

a

$$\boxed{s = a \times b \bmod q}$$

b

$c_1 = E_A(a)$

$$\xrightarrow{\quad c_1 \quad}$$

$\longrightarrow m$

$$\xleftarrow{\quad c_2 \quad}$$

$c_2 = c_1 \otimes b \oplus m = E_A(ab + m)$

$a' = D_A(c_2)$

$b' = -m$

$$\boxed{a' + b' = (ab + m) + (-m) = a \times b = s}$$

Paillier Modulus

We will choose the Paillier modulus N large enough so that operations modulo N will not “wrap around” and will be consistent to doing them over the integers.

However ...

- If *a, b, m* are in Z_q and $N > q^3$ protocol will work
- Players can maliciously choose their values to be larger
  - Protocol will fail, but failure may reveal information about the honest players' input
- Two options
  - Expensive: Include a range proof. No additional assumptions
  - Cheaper: No range proof. Assume that information leaked will not help forging DSA signatures

GMW product

$a = a_1 + a_2 + \ldots + a_n$

$b = b_1 + b_2 + \ldots + b_n$

a_1 , b_1

a_2 , b_2

a_3 , b_3

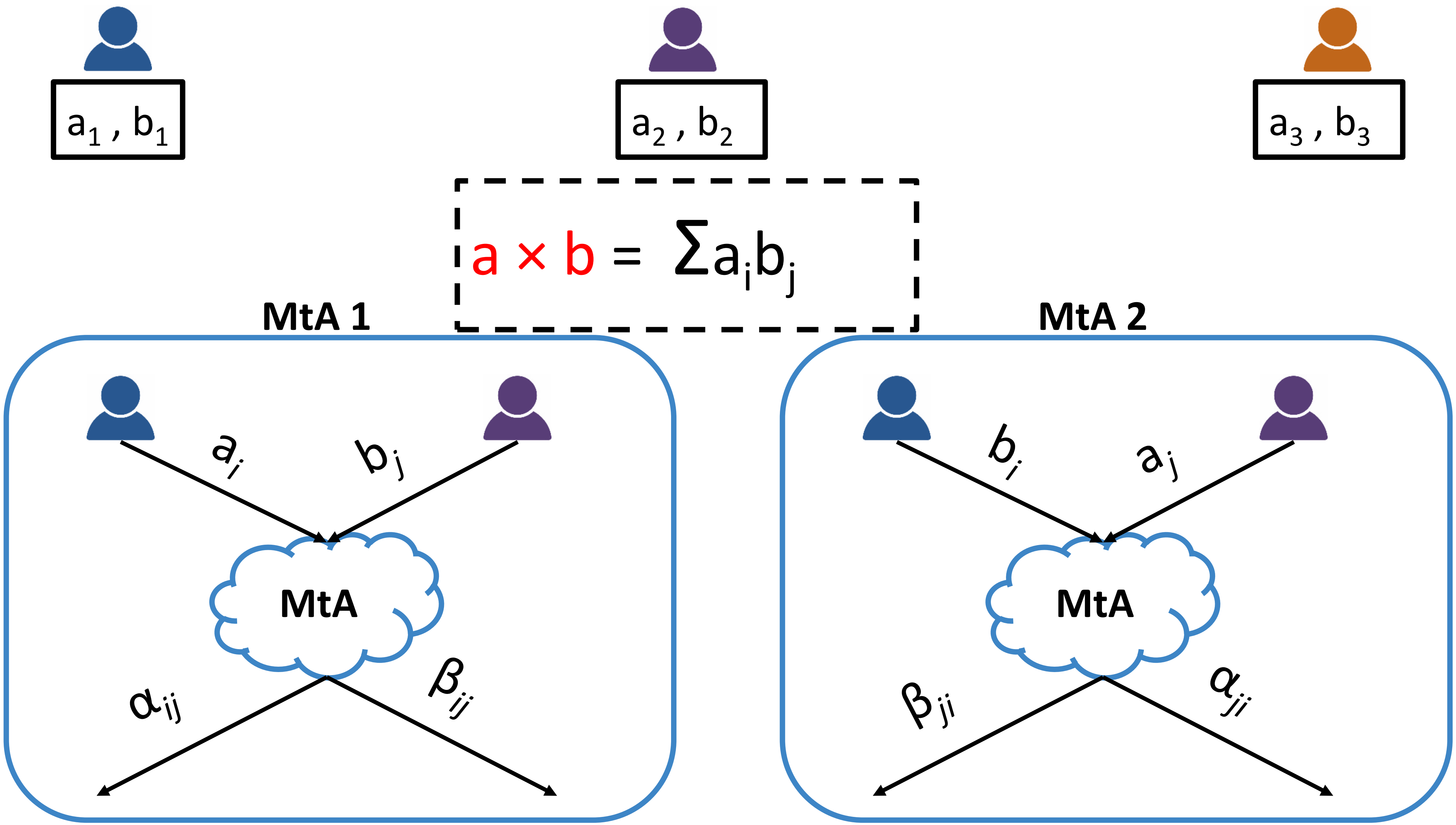
$$a \times b = \Sigma a_i b_j$$

P_i engages in two (2) MtA protocols with every other party P_j

GMW product

$a = a_1 + a_2 + \ldots + a_n$

$b = b_1 + b_2 + \ldots + b_n$



Sharing a product

$a = a_1 + a_2 + \ldots + a_n$

$b = b_1 + b_2 + \ldots + b_n$

a_1 , b_1

a_2 , b_2

a_3 , b_3

$$a \times b = \sum a_i b_j$$

P_i's share is

$$a_i b_i + \sum_j (\alpha_{ij} + \beta_{ji})$$

Threshold ECDSA from MtA

Key generation

- Players distributedly generate Shamir shares of a secret key x
  - Each player contributes randomness to x and distributes shares to all other players
- Each players ends up with a key share x_i
- Everyone learns public key $y = g^x$

Computing $R=g^k$

- Beaver's trick
- Distributively generate shared random values k and γ
  - Every player has shares k_i and γ_i
- Use MtA to get additive shares δ_i of $\delta = k\gamma$
- Reveal δ and g^k
  - via interpolation and interpolation in the exponent respectively
- Each player sets $t_i=\delta^{-1}\gamma_i$
  - the t_i interpolate to k^{-1}

Computing $s=k^{-1}(m+xr)$

- Use MtA protocol on shares of k^{-1} and x
  - End up with shares s_i of s

Cannot publish s_i until checking that the signature is correct

The problem

- Adversary might have not inputted correct values in the MtA protocols

- Shares of s are now incorrect
  - Players could detect that by checking if the signature actually verifies or not
  - But the incorrect share held by the good players may reveal information

- Solution: randomize the shares so that
  - if they are correct the signature verifies
  - if they are incorrect the shares of good players are mapped to random points

Distributed validity test

- $\mathbf{R^s = g^{-m}\, y^{-r}}$
- Each player reveals $\mathbf{R^{si}}$ masked by $\mathbf{g^{li}}$
  - $\mathbf{V_i = R^{si}\, g^{li}}$
- $\mathbf{V = g^{-m}\, y^{-r}}$ **Prod** $\mathbf{V_i}$ should be $\mathbf{g^l}$
- Players can check that via a distributed Diffie-Hellman
  - Broadcast $\mathbf{A_i = g^{ri}}$
    - $\mathbf{A =}$ **Prod** $\mathbf{A_i = g^r}$
  - Broadcast $\mathbf{T_i = A^{li}}$ and $\mathbf{U_i = V^{ri}}$
    - **Prod** $\mathbf{T_i}$ should be equal to **Prod** $\mathbf{U_i}$ (both $\mathbf{g^{lr}}$)
    - pseudorandom values if test fails (under DDH)

Security Proof & Extensions

- Main proof in the paper is in the game-based definition of security
  - It is hard to forge DSA signatures even if controlling t players
- Simulation based proof is possible for our protocol if players prove knowledge of their inputs to **all** MtA protocols
  - does not have to be range proofs necessarily
- MtA protocol is used as a black box
  - can use any, including the OT based one by Gilboa in the malicious adversary version presented earlier
- Open source implementation by KZen Networks
  - https://github.com/KZen-networks/multi-party-ecdsa