ID-based Digital Signature Algorithms

Jerome A. Solinas

National Security Agency

ECC ‘03

ID-based Cryptography

- Public key = User's ID
- Don't need certificates to authenticate
- Based on ***bilinear pairings*** on elliptic curves

Bilinear Pairings

The pairing $<P,Q>$ of the order-r points P and Q is an r^{th} root of 1

$$< P, Q_0 + Q_1 > = < P, Q_0 > < P, Q_1 >$$

$$< P_0 + P_1, Q > = < P_0, Q > < P_1, Q >$$

$$< kP, Q > = < P, kQ > = < P, Q >^k$$

Key Agreement

(Sakai-Ohgishi-Kasahara, 2000)

s	systemwide **master key**
C_i	user i's public key (based on ID_i)
$V_i = s\,C_i$	user i's private key

$$SSV = \langle V_A, C_B \rangle = \langle C_A, V_B \rangle$$

(Diffie-Hellman analogue)

One-Way Key Agreement

G	order - r *base point*
$J = s\,G$	*universal public key*
z_i	one-time private key
$Y_i = z_i\,G$	one-time public key

Alice computes $\mathrm{SSV}_{A,B} = < z_A C_B , J >$

Bob computes $\mathrm{SSV}_{A,B} = < V_B , Y_A >$

Public-Key Encryption

(Boneh-Franklin, 2001)

Session Key

$$\mathcal{K} = \mathrm{KDF}(\mathrm{SSV}_{A,B})$$

Encrypted Message

$$(Y_A , \mathrm{Enc}_{\mathcal{K}}(\mathcal{M}))$$

(ElGamal analogue)

Authenticated Key Agreement

(Smart, 2002)

Alice sends Bob $SSV_{A,B}$

Bob sends Alice $SSV_{B,A}$

$SSV = SSV_{A,B} \cdot SSV_{B,A}$

(MQV analogue)

More Accurate Picture

E	defined over GF(q)
r	prime divisor of $\#E(\mathrm{GF}(q))$
G	order r point in $E(\mathrm{GF}(q))$
$\mathcal{G}$	multiples of G
d	small *extension degree*
H	order r point in $E(\mathrm{GF}(q^d))$
$\mathcal{H}$	multiples of H

Typical pairing: $< P, Q >$, $P \in \mathcal{G}$, $Q \in \mathcal{H}$

More Accurate Picture (cont'd)

s	master key
$C_i \in \mathcal{G}$	user i's public key
$V_i = sC_i \in \mathcal{G}$	user i's private key
$G \in \mathcal{G}, \ H \in \mathcal{H}$	base points
$J=sG, \ K=sH$	universal public keys
z_i	one-time private key
$Y_i = z_i G, \quad Z_i = z_i H$	one-time public keys

Digital Signatures

$\mathcal{M}$	message to be signed by Alice
sig $\mathcal{M}$	message signature (sent along with $\mathcal{M}$)
$\mathcal{H}$	Cryptographic hash function

Digital Signature #1

(Sakai-Kasahara, 2003: ElGamal analogue)

z_A, Z_A	one-time key pair
x	x coordinate of Z_A
h	$\mathsf{H}(\mathcal{M})$
S	$z_A^{-1} h\, C_A + z_A^{-1} x\, V_A$
sig $\mathcal{M}$	$(Z_A\, ,\, S)$

Verify:

$$< S, Z_A > \stackrel{?}{=} < C_A\, ,\, h\, H + x\, K >$$

Digital Signature #2

(Sakai-Kasahara, 2003: Schnorr analogue)

z_A, Z_A	one-time key pair
e	$< C_A , Z_A >$
h	$\mathsf{H}(\mathcal{M}, e)$
S	$h\, V_A + z_A\, C_A$
sig $\mathcal{M}$	(h , S)

Verify:

$$e = < S , H > < C_A , -h\, K >$$

$$\mathsf{H}(\mathcal{M}, e) \stackrel{?}{=} h$$

Digital Signature #3

(Sakai-Ohgishi-Kasahara, 2000)

z_A, Z_A	one-time key pair
R	embedding of $\mathcal{M}$ into $\mathcal{G}$
S	$V_A + z_A R$
sig $\mathcal{M}$	(Z_A , S)

Verify:

$$< S, H > \stackrel{?}{=} < C_A , K > < R , Z_A >$$

Digital Signature #4

(Paterson, 2002)

z_A, Z_A	one-time key pair
h_0	$\mathrm{H}(\mathcal{M})$
h_1	$\mathrm{H}(Z_A)$
S	$z_A^{-1} h_0 G + z_A^{-1} h_1 V_A$
sig $\mathcal{M}$	(Z_A , S)

Verify:

$$< S, Z_A > \stackrel{?}{=} < G , H >^{h_0} < C_A , K >^{h_1}$$

Note: $< G, H >$ is a systemwide constant.

Digital Signature #5

(Cha-Cheon, 2002)

z_A	one-time private key
S_0	$z_A\, C_A$
h	$H(\mathcal{M}, S_0)$
S_1	$(z_A + h)\, V_A$
sig $\mathcal{M}$	(S_0, S_1)

Verify:

$$< S_0 + h\, C_A, K > \stackrel{?}{=} < S_1, H >$$

Digital Signature #6

(Xun Yi, 2003)

z_A, Y_A	one-time key pair
h	$H(\mathcal{M}, Y_A)$
S	$z_A J + h V_A$
sig $\mathcal{M}$	(Y_A, S)

Verify:

$$< S , H > \stackrel{?}{=} < Y_A + h C_A , K >$$

Cost of Signature Algorithms

Depends on implementation choices:

- Choice of field: characteristic 2, 3 or (large) *p* [We choose *p* for exposition]

- Type of elliptic curve: *ordinary* vs. *supersingular* [Affects signature size]

- Type of pairing: Weil vs. Tate [Affects verification cost]

Ordinary & Supersingular Curves

Supersingular curves: $d=2$

Example: $|q|=512$, $|q^2|=1024$, $|r|=160$

Ordinary curves: $d=6$

Example: $|q|=170$, $|q^6|=1024$, $|r|=160$

Point Sizes

(above example, cont'd)

- Ordinary curves:

  171 bits for $\mathcal{G}$, 1025 bits for $\mathcal{H}$

- Supersingular curves: if $Q \in \mathcal{H}$, then $Q = \phi(P)$ (ϕ = *distortion*) for some $P \in \mathcal{G}$

  513 bits for $\mathcal{G}$, 513 bits for $\mathcal{H}$

Signature Sizes

(above example, cont'd)

Signature	Ordin.	Supersing.
SK-ElGamal	1196	1026
SK-Schnorr	331	673
S-O-K	1196	1026
Paterson	1196	1026
Cha-Cheon	342	1026
XY	342	1026

Embedding

• *Embedding* a bit string means finding a corresponding element of $\mathcal{G}$ according to a fixed rule.

• Most expensive part: ($|q|$-$|r|$)-bit elliptic scalar multiplication.

• (above example) requires 10-bit ESM for E ordinary, 352-bit ESM for E supersingular.

• *Example:* constructing public key C_i from ID_i.

Weil and Tate Pairings

$M(P,Q)$ = Miller expression

$W(P,Q) = M(P,Q) / M(Q,P)$

$T(P,Q) = M(P,Q)^c, \quad c=(q^d - 1)/r$

$W(P,Q)$ and $T(P,Q)$ bilinear

$W(P,Q) = W(Q,P)^{-1} \qquad \therefore \; W(P,P)=1$

Relative Costs

(increasing order of cost)

- Modular exponentiation
- Elliptic scalar multiplication
- "Miller Lite" $M(P,Q)$, $P \in \mathcal{G}$
- general Miller expression

so: choose Weil vs. Tate to minimize number of Miller evaluations

Weil vs. Tate Pairings

$W(P,Q) = M(P,Q) / M(Q,P)$

cost: 1 Miller lite + 1 Miller

$T(P,Q) = M(P,Q)^c$

cost: 1 Miller lite + 1 modular expon.

so: a single Tate pairing is cheaper than a single Weil pairing

Products of Pairings

Let $P, P' \in \mathcal{G}$ and $Q, Q' \in \mathcal{H}$

$T(P,Q)\, T(P',Q') = (M(P,Q)\, M(P',Q'))^c$

cost: 2 Miller lites + 1 modular expon.

$W(P,Q)\, W(P',Q') = W(P\text{-}Q', P'+Q)$

cost: 2 Millers

Precomputation Costs

Signature	Precomp cost
SK-ElGamal	2 ESM's + 1 inversion (mod r)
SK-Schnorr	1 ESM + 1 modular exp. lite
S-O-K	1 ESM
Paterson	2 ESM's + 1 inversion (mod r)
Cha-Cheon	1 ESM
XY	1 ESM

Signing Costs

Signature	Signing cost
SK-ElGamal	1 ESM
SK-Schnorr	1 ESM
S-O-K	1 ESM + 1 Embedding
Paterson	1 ESM
Cha-Cheon	1 ESM
XY	1 Twin ESM

Verification Costs

M=Miller S=ESM E=Expon.

ML=Miller Lite Tw=Twin op. EL=Exp. Lite

Signature	Weil	Tate
SK-ElGamal	2M+TwS	2ML+TwS+E
SK-Schnorr	2M+S	2ML+E+EL
S-O-K	4M	3ML+E
Paterson	2ML+S+EL	2ML+TwE+EL
Cha-Cheon	2M+S	2ML+S+E
XY	2M+S	2ML+S+E

Per-User Constants

- The quantities

$$\gamma = < C_A , K > \quad \text{and} \quad \tau = < C_A , H >$$

can be computed and stored and reused for repeated verifications from the same signer.

- *Example:* Single XY (WP) verification

$$< S + K , H + D_A + h\, C_A > \stackrel{?}{=} 1$$

Repeated verification

$$< S + K , H + D_A > \stackrel{?}{=} \gamma^h$$

Cost of Repeated Verifications

Tw=Twin op. Tr=Triple op.

Signature	Weil	Tate
SK-ElGamal	2ML+TwEL	ML+TrE
SK-Schnorr	2ML+EL	ML+TwE
S-O-K	2M	2ML+E
Paterson	2ML+TwEL	ML+TrE
Cha-Cheon	2M+EL	2ML+TwE
XY	2M+EL	2ML+TwE

Which signature is best?

Signature length (E ord.)	Schn. (CC/XY)
Signature length (E sup.)	Schn.
Signing cost	ElG/Schn/Pat/CC
Single verif. cost (WP)	Pat *or* Schn/CC/XY
Single verif. cost (TP)	Schn.
Repeat verif. cost (WP)	SOK *or* Schn.
Repeat verif. cost (TP)	Schn.

Best Pairing-Based Algorithms?

Real answer: depends on

- What field? (char = 2, 3, or p?)
- E ordinary or supersingular?
- Weil or Tate pairing?
- Exact choices for $|r|$ and $|q|$
- Relative costs of exponentiation, elliptic scalar multiplication, Miller computation, (single, twin, triple, and lite versions)
- Ability to store (or send) per-user constants