Pairing-Based Cryptography

Sanjit Chatterjee

Indocrypt 2010

Indian Institute of Science

Taj Krishna *to* Courtyard Marriott

Indocrypt'02 Hyderabad:
Programme Co-chairs: Alfred Menezes (UW) and Palash Sarkar (ISI).

Taj Krishna *to* Courtyard Marriott

Indocrypt'02 Hyderabad:
Programme Co-chairs: Alfred Menezes (UW) and Palash Sarkar (ISI).

Tutorial: Constructive applications of the Weil and Tate pairings. Speaker: Alfred Menezes.

Taj Krishna *to* Courtyard Marriott

Indocrypt'02 Hyderabad:
Programme Co-chairs: Alfred Menezes (UW) and Palash Sarkar (ISI).

Tutorial: Constructive applications of the Weil and Tate pairings. Speaker: Alfred Menezes.

Today's tutorial is based on what I learned from Palash and Alfred (and my other teachers and colleagues at ISI and UW).

In a nutshell

"Tutorials provide a wonderful way to share confusion and can convince the attendants that they are not the only dumb people out there."

In a nutshell

“Tutorials provide a wonderful way to share confusion and can convince the attendants that they are not the only dumb people out there.”

Feel free to have your *siesta*...
But *don’t forget* to put your mobile friend in the sleep mode.

“Pairings for Cryptographers”

Many research papers in the field treat pairings as a **black box** *and then proceed to build various cryptographic schemes making use of assumed properties of the pairings. This is not necessarily a bad approach...*

“Pairings for Cryptographers”

Many research papers in the field treat pairings as a **black box** *and then proceed to build various cryptographic schemes making use of assumed properties of the pairings. This is not necessarily a bad approach...*

However, if this approach is taken, then it is easy for authors to make assumptions concerning the properties of pairings which are not necessarily correct, and hence develop cryptographic schemes which cannot be realized in practice, or which cannot be implemented as efficiently as the authors assume.

Our Agenda

Take a panoramic view of the interplay of functionality, security and efficiency of cryptographic protocols employing bilinear pairing.

Pairing in Cryptology

“New Directions in Cryptography”

- Diffie-Hellman (1976)
  - The (public) birth of Public-Key Cryptography.
- Two-party one-round key agreement protocol.
  - A prime order group $\mathbb{G} = \langle P \rangle$
  - $\hat{A} \rightarrow \hat{B} : aP$.
  - $\hat{B} \rightarrow \hat{A} : bP$.
  - Shared secret: abP.
- A *natural* question:
  - Design a three-party one-round key agreement protocol.

Intractability Assumptions

- Security of any public key cryptographic protocol is based on the assumed *hardness* of some computational problem.
  - Unconditional proof of security of a public-key protocol will imply $P \neq NP$.
- Discrete logarithm problem (DLP).
  - A classical problem in number theory.
  - Given $\mathbb{G} = \langle P \rangle$ and $R \in \mathbb{G}$, find a such that $R = aP$.
- Security of Diffie-Hellman protocol is based on the Diffie-Hellman Problem (DHP).
  - Given $\langle P \rangle, aP, bP$, compute abP.

Elliptic Curve Cryptography

- Proposed (independently) by Koblitz and Miller (1985).
- No faster-than-squareroot generic algorithm to solve DLP over Elliptic Curve groups.
- Several curve forms were suggested for cryptographic use.
    - "Supersingular curves" was one of them.

Pairing: a *black box* intro

- Let n be a prime number.
- Let $\mathbb{G} = \langle P \rangle$ be an additive group of order n.
- Let $\mathbb{G}_T$ be a multiplicative group of order n.

A *symmetric* bilinear pairing on $(\mathbb{G},\ \mathbb{G}_T)$ is a function $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ such that:

1. Bilinearity: For all $R, S, T \in \mathbb{G}$:

$$e(R + S, T) = e(R, T)e(S, T)$$

$$e(R, S + T) = e(R, S)e(R, T)$$

2. Non-degeneracy: $e(P, P) \neq 1$.

Known examples: Weil pairing, Tate pairing over *supersingular* elliptic curves.

Pairing in crypto

$$\boxed{e(aP, bP) = e(P, P)^{ab}}$$

- Menezes-Okamoto-Vanstone (1991)
    - Reducing elliptic curve logarithms to logarithms in a finite field.
- The MOV reduction:
    - $\mathrm{DLP}_{\mathbb{G}}$: Given $P, aP \in \mathbb{G}$, find a.
    - Compute $\alpha = e(P, P)$, $\beta = e(P, aP) = \alpha^a$.
        - $a = \log_\alpha \beta$.
- Weil pairing was employed.
    - Around 16 minutes to compute the pairing.
- Frey-Ruck (1994): used Tate pairing to reduce ECDLP to DLP in finite field.

A curse for ECC?

$$\boxed{e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T}$$

- Use faster algorithm in $\mathbb{G}_T$ to solve the original DLP in $\mathbb{G}$.
- Could be a problem for ECC if the parameters are not appropriately chosen.
- Supersingular curves were (almost) abandoned...if not ECC altogether.
  - See [Koblitz-Koblitz-Menezes:2008] for the history of the acceptance of ECC.

CDH *vs.* DDH

$$\boxed{e(aP, bP) = e(P, P)^{ab}}$$

- MOV/FR reduction: a partial utilization of the *magic* of bilinearity!
- Decision Diffie-Hellman problem in $\mathbb{G}$:
  - Given (P, aP, bP, cP) decide whether $c = ab$.

CDH *vs.* DDH

$$\boxed{e(aP, bP) = e(P, P)^{ab}}$$

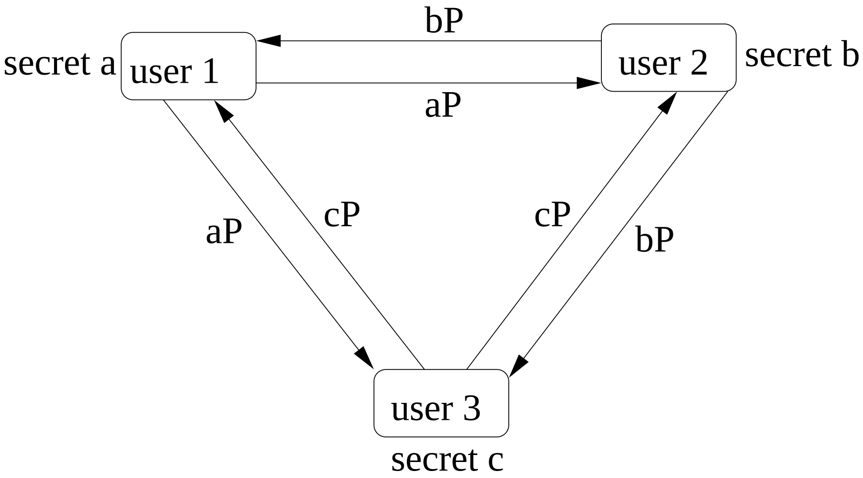
- MOV/FR reduction: a partial utilization of the *magic* of bilinearity!
- Decision Diffie-Hellman problem in $\mathbb{G}$:
  - Given (P, aP, bP, cP) decide whether $c = ab$.
- DDH is easy in $\mathbb{G}$ if one can efficiently compute the pairing:

$$e(aP, bP) \stackrel{?}{=} e(P, cP).$$

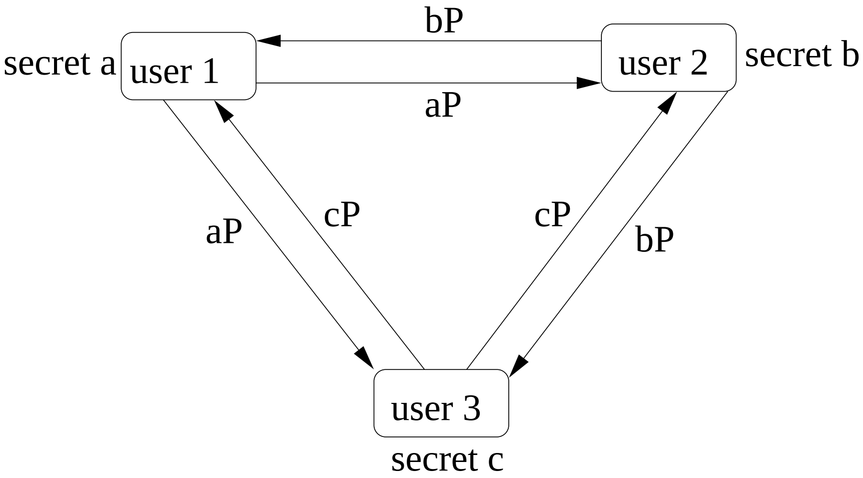
Come 2000!

- Sakai-Ohgishi-Kasahara ID-based key agreement protocol [SCIS'00].
- Joux's three-party one round key agreement protocol [ANTS'00].
- Constructive use of pairing to solve two interesting cryptographic problems.

Joux's Protocol



Joux's Protocol



$$K = e(P,P)^{abc} = e(bP,cP)^{a} = e(aP,cP)^{b} = e(aP,bP)^{c}.$$

A blessing for cryptographers!

- Identity-Based Encryption (2001).
  - Boneh and Franklin solved the open problem posed by Shamir in 1984.
  - Presented in the language of provable security.
  - Cited more than 3500 times.
- Short signature (2001).
- Hierarchical IBE (2002).
- Aggregate signatures, homomorphic encryption, broadcast encryption, attribute-based encryption...
- Non-interactive zero knowledge proofs...
- Pairing – a new conference starting in 2007
  - *Pairing 2010:* Dec. 13 to 15, 2010.

Identity-Based Cryptography

ID-based Non-interactive Key Distribution

- A and B with id $\mathsf{ID_A}$ and $\mathsf{ID_B}$ agree upon a common key without *any* interaction among themselves.

ID-based Non-interactive Key Distribution

- A and B with id ID_A and ID_B agree upon a common key without *any* interaction among themselves.
- A *Private Key Generator* (PKG) provides the private key corresponding to each identity.

ID-NIKD based on pairing

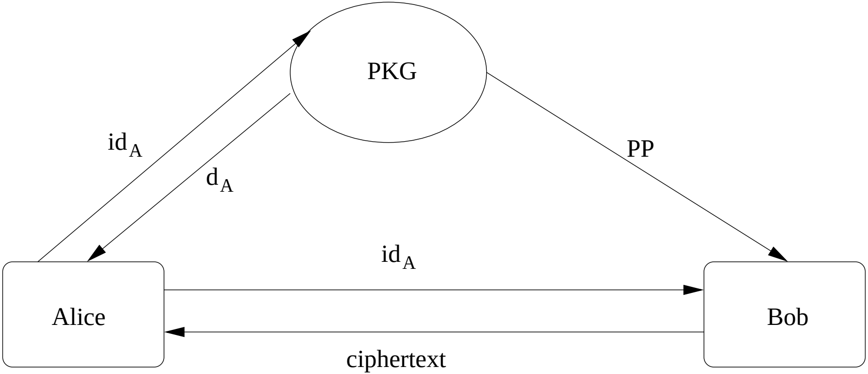
[Sakai-Ohgishi-Kasahara:2000]

- $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, $\mathbb{G} = \langle P \rangle$.
- $H : \{0,1\}^* \to \mathbb{G}$ is a hash function.
- PKG chooses master secret $s \in_R \mathbb{Z}_n$ and sets public key $R = sP$.

Key Extraction: The pvt. key d_A of user A with identity ID_A is $d_\mathrm{A} = sQ_\mathrm{A}$, where $Q_\mathrm{A} = H(\mathrm{ID}_\mathrm{A})$.
Key Agreement: A computes $K_A = e(d_A, Q_B) = e(Q_A, Q_B)^s$ and B computes $K_B = e(d_B, Q_A) = e(Q_B, Q_A)^s$.

Note: $e(Q_A, Q_B) = e(Q_B, Q_A)$ and hence $K_A = K_B$ is the shared secret.

Identity-Based Encryption (IBE)



IBE (a little more formal)

Set-Up: The algorithm returns the system public parameters PP together with the master secret key msk.

Key-Gen: Takes input an identity $\mathrm{id} \in \mathcal{I}$ together with PP and msk and returns a pvt. key d_{id}.

Encrypt: Takes input an identity $\mathrm{id} \in \mathcal{I}$, a message $M \in \mathcal{M}$ and PP and produces a ciphertext $C \in \mathcal{C}$.

Decrypt: Takes as input $C \in \mathcal{C}$, id and a corresponding private key d_{id}, PP. It returns the message M or $\perp$ if the ciphertext cannot be decrypted.

These set of algorithms must satisfy the standard soundness requirement.

BasicIdent of Boneh-Franklin (simplified)

$e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, $\mathbb{G} = \langle P \rangle$.
$H : \{0,1\}^* \to \mathbb{G}^*$ is a hash function.

- Set-Up: msk: $s \in_R \mathbb{Z}_n^*$ and set $P_{\text{pub}} = sP$.
- Key-Gen: Given ID $\in \{0,1\}^*$, compute $Q_{\text{ID}} = H(\text{ID})$ and set $d_{\text{ID}} = sQ_{\text{ID}}$.
- Encrypt: To encrypt $M \in \mathbb{G}_T$ to ID compute $Q_{\text{ID}} = H(\text{ID})$, choose $r \in_R \mathbb{Z}_n^*$ and set:

$$C = \langle rP, M \times e(Q_{\text{ID}}, P_{\text{pub}})^r \rangle$$

- Decrypt: To decrypt $C = \langle U, V \rangle$ using d_{ID} compute

$$V \times e(d_{\text{ID}}, U)^{-1} = M.$$

$$e(d_{\text{ID}}, U) = e(sQ_{\text{ID}}, rP) = e(Q_{\text{ID}}, sP)^r = e(Q_{\text{ID}}, P_{\text{pub}})^r.$$

BasicIdent from SOK ID-NIKD

BasicIdent from SOK ID-NIKD

- Set-up and Key-Gen algorithms are same.
- Encrypt and Decrypt of BasicIdent can be obtained by slightly tweaking the Key Agreement algorithm.
    - Encrypt masks the message by $e(P_{\text{pub}}, Q_{\text{ID}})^r = e(rsP, Q_{\text{ID}}) = e(s(rP), Q_{\text{ID}})$ and rP is sent as part of C.
    - Recipient with pvt. key $d_{\text{ID}} = sQ_{\text{ID}}$ unmasks by computing $e(rP, d_{\text{ID}}) = e(s(rP), Q_{\text{ID}}) = e(rP, Q_{\text{ID}})^s$.
    - Think of rP as the hash of id of the sender.
    - The shared secret in SOK ID-NIKD is transformed into the mask in C of BasicIdent.

Security Model for IBE

Security Model for IBE

A game between an adversary ($\mathcal{A}$) and a challenger ($\mathcal{B}$).

- $\mathcal{A}$ can place
  - Key-extraction query: adaptively corrupt users.
  - Decryption query: obtain the decryption of a ciphertext under an id of its choice.
- $\mathcal{A}$ chooses a target ID^* and two messages M_0 and M_1.
  - Receives an encryption of M_β, $\beta \in_R \{0,1\}^*$ under ID^*.
- Can continue with the key-extraction and decryption queries.
- $\mathcal{A}$ wins if it can predict β with prob. significantly away from $1/2$.

CPA-security: Decryption queries are disallowed.

Security of BasicIdent

- Bilinear Diffie-Hellman Problem (BDH): Given $\langle P, aP, bP, cP \rangle$ for unknown $a, b, c \in \mathbb{Z}_p$, compute $e(P, P)^{abc}$.
- Decision Bilinear Diffie-Hellman Problem (DBDH): Given $\langle P, aP, bP, cP, Z \rangle$ for unknown $a, b, c \in \mathbb{Z}_p$, decide whether $Z = e(P, P)^{abc}$ or Z is a random element of $\mathbb{G}_T$.
- *Security* of BasicIdent is based on the hardness of (D)BDH problem.
  - Given $\mathcal{A}$ against BasicIdent construct an algorithm $\mathcal{B}$ that solves the DBDH problem.
  - $\mathcal{A}$ can corrupt users (other than the target).
  - But cannot place any decryption query.

Reductionist Argument (simplified)

- DBDH instance: $\langle P, aP, bP, cP, Z\rangle$.
- Set $P_{\text{pub}} = aP$.
- Fix the target ID^*.
- H-query:
  - For $\text{ID} \neq \text{ID}^*$, choose $x \in_R \mathbb{Z}_p^*$ and set $H(\text{ID}) = xP$.
  - Set $H(\text{ID}^*) = bP$.
- Pvt. key query on ID: return $x(aP)$.
- Given M_0, M_1, choose $\beta \in_R \{0, 1\}$ and set
  - $C = \langle cP, M_\beta \times Z\rangle$.
- If $Z = e(P, P)^{abc}$ then C is a proper ciphertext.
- Otherwise C statistically hides β.
- Advantage of $\mathcal{A}$ against BasicIdent can be converted into an advantage to solve the DBDH instance.

CCA Security

- BasicIdent: insecure *if* $\mathcal{A}$ can obtain decryption of ciphertexts of its choice.
    - Given $C = \langle rP, M_\beta \times e(P_{\text{pub}}, Q_{\text{ID}})^r \rangle$.
    - $\mathcal{A}$ asks for the decryption of

$$C' = \langle rP + r'P, M_\beta \times e(P_{\text{pub}}, Q_{\text{ID}})^r \times e(P_{\text{pub}}, Q_{\text{ID}})^{r'} \rangle.$$

    - Receives M_β!
- FullIdent: CCA-secure version of Boneh-Franklin IBE.
    - Adapts the Fujisaki-Okamoto transformation to IBE.

Another look at BF-IBE

- [Galindo:2005] The security argument of FullIdent is flawed.
  - The flaw creeps in due to ciphertext integrity check in decrypt.
  - Transmits to several extensions of BF-IBE.
  - Fortunately the flaw can be fixed.
- Galindo describes the protocol in *asymmetric* pairing setting.
  - $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$.
  - Helps to reduce the ciphertext overhead.
- Assumes
  - One can hash efficiently into $\mathbb{G}_2$.
  - An efficiently computable isomorphism $\psi : \mathbb{G}_2 \rightarrow \mathbb{G}_1$.

Another look at BF-IBE

- [Galindo:2005] The security argument of FullIdent is flawed.
  - The flaw creeps in due to ciphertext integrity check in decrypt.
  - Transmits to several extensions of BF-IBE.
  - Fortunately the flaw can be fixed.
- Galindo describes the protocol in *asymmetric* pairing setting.
  - $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$.
  - Helps to reduce the ciphertext overhead.
- Assumes
  - One can hash efficiently into $\mathbb{G}_2$.
  - An efficiently computable isomorphism $\psi : \mathbb{G}_2 \rightarrow \mathbb{G}_1$.
- Unfortunately both cannot be achieved simultaneously! ...more about this later.

Boneh-Lynn-Shacham Signature [2001]

- IBE to Signature: Naor's observation
  - The private key corresponding to an ID can serve as a signature on a message (think ID as the message to be signed).
- BasicIdent yields BLS signature.
- The signature is an element of $\mathbb{G}$.

Boneh-Lynn-Shacham Signature [2001]

- IBE to Signature: Naor's observation
  - The private key corresponding to an ID can serve as a signature on a message (think ID as the message to be signed).
- BasicIdent yields BLS signature.
- The signature is an element of $\mathbb{G}$.
- But we already have practical signature schemes.
  - Can we have a short(er) signature?

Asymmetric Pairings

(Asymmetric) Pairings: (still) in abstract

Let $\mathbb{G}_1 = \langle P_1 \rangle$, $\mathbb{G}_2 = \langle P_2 \rangle$ and $\mathbb{G}_T$ be groups of order n with $\mathbb{G}_1 \neq \mathbb{G}_2$.

An *asymmetric* bilinear pairing on $(\mathbb{G}_1, \mathbb{G}_2,\ \mathbb{G}_T)$ is a function $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ such that:

1. Bilinearity: For all $Q_1, Q_2 \in \mathbb{G}_1,\ R_1, R_2 \in \mathbb{G}_2$:

$$e(Q_1 + Q_2, R_1) = e(Q_1, R_1)e(Q_2, R_1)$$

$$e(Q_1, R_1 + R_2) = e(Q_1, R_1)e(Q_1, R_2).$$

2. Non-degeneracy: $e(P_1, P_2) \neq 1$.

Note: $e(aU, bV) = e(U, V)^{ab} = e(bU, aV)\ \ \forall U \in \mathbb{G}_1, V \in \mathbb{G}_2, a, b \in \mathbb{Z}$.
Known examples: Weil pairing, Tate pairing over *ordinary* elliptic curves.

Why Asymmetric Pairings?

- Symmetric pairings are restricted in the choice of curves.
- And *significantly* slower at higher security levels.
- Asymmetric pairings allow **shorter** representation of elements of $\mathbb{G}_1$.

BLS *Short* Signature

[Boneh-Lynn-Shacham:2001]

- **Key generation:** Private key: $x \in_R \mathbb{Z}_n^*$ and Public key: $X = xP_2 \in \mathbb{G}_2$.
- **Sign:** To sign $M \in \{0,1\}^*$:
  - Compute $H = \mathrm{Hash}(M)$, where $\mathrm{Hash} : \{0,1\}^* \to \mathbb{G}_1$.
  - Compute $\sigma = xH \in \mathbb{G}_1$.

  The signature on M is σ.
- **Verify:** To verify (M, σ):
  - Compute $H = \mathrm{Hash}(M)$.
  - Accept iff $e(\sigma, P_2) = e(H, X)$.

Correctness: $e(\sigma, P_2) = e(xH, P_2) = e(H, xP_2) = e(H, X)$.

Security definition

Security definition

- *Existential Unforgability under Chosen Message Attack.*
- A game between a challenger ($\mathcal{B}$) and an attacker $\mathcal{A}$.
    - $\mathcal{A}$ is provided with the public key and a *signing oracle*.
    - $\mathcal{A}$ can adaptively ask for the signature on any message M of its choosing.
    - $\mathcal{A}$'s task is to produce a valid message-signature pair (M^*, σ^*).
- *Security* is established through a reductionist argument.
    - If $\mathcal{A}$ is successful then $\mathcal{B}$ solves some computational problem which is *believed* to be hard.

BLS Security

co-DHP in $(\mathbb{G}_1, \mathbb{G}_2)$: Given $H \in \mathbb{G}_1$ and $X (= xP_2) \in \mathbb{G}_2$, compute xH.

Theorem: If co-DHP in $(\mathbb{G}_1, \mathbb{G}_2)$ is hard and Hash is a *random* function, then the BLS-2 signature scheme is secure.

Argument: Given $X \in \mathbb{G}_2$:

- The signing oracle is *useless*:
  - Gives $\sigma \in \mathbb{G}_1$ s.t. $e(\sigma, P_2) = e(H', X)$ for $H' \in_R \mathbb{G}_1$.
  - *But* $\mathcal{A}$ can generate such (H', σ) pair itself – select $y \in_R \mathbb{Z}_n^*$ and compute $H' = yP_1$ and $\sigma = y\psi(X) = yxP_1 = xH'$.
- So $\mathcal{A}$'s problem is to compute $\sigma = xH$, given $H \in \mathbb{G}_1$ and $X \in G_2$.
  - This is precisely co-DHP in $(\mathbb{G}_1, \mathbb{G}_2)$.

BLS Security

co-DHP in $(\mathbb{G}_1, \mathbb{G}_2)$: Given $H \in \mathbb{G}_1$ and $X(= xP_2) \in \mathbb{G}_2$, compute xH.

Theorem: If co-DHP in $(\mathbb{G}_1, \mathbb{G}_2)$ is hard and Hash is a *random* function, then the BLS-2 signature scheme is secure.

Argument: Given $X \in \mathbb{G}_2$:

- The signing oracle is *useless*:
  - Gives $\sigma \in \mathbb{G}_1$ s.t. $e(\sigma, P_2) = e(H', X)$ for $H' \in_R \mathbb{G}_1$.
  - *But* $\mathcal{A}$ can generate such (H', σ) pair itself – select $y \in_R \mathbb{Z}_n^*$ and compute $H' = yP_1$ and $\sigma = y\psi(X) = yxP_1 = xH'$.
- So $\mathcal{A}$'s problem is to compute $\sigma = xH$, given $H \in \mathbb{G}_1$ and $X \in G_2$.
  - This is precisely co-DHP in $(\mathbb{G}_1, \mathbb{G}_2)$.

What is ψ??

Type 2 and Type 3 Pairings

$$\boxed{e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T}$$

- If an efficiently-computable isomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$ $(\psi(P_2) = P_1)$, is known, then e is called a Type 2 pairing.
- If no such isomorphism ψ is known, then e is called a Type 3 pairing.

- BLS-2: BLS employing Type 2 pairing.
- BLS-3: BLS employing Type 3 pairing.

BLS-3 Security

co-DHP* in $(\mathbb{G}_1, \mathbb{G}_2)$: Given $H, xP_1(= X_1) \in \mathbb{G}_1$ and $xP_2(= X_2) \in \mathbb{G}_2$, compute xH.

Theorem: If co-DHP* in $(\mathbb{G}_1, \mathbb{G}_2)$ is hard and Hash is a *random* function, then the BLS-3 signature scheme is secure.

Argument: Given $X_1 \in \mathbb{G}_1$ and $X_2 \in \mathbb{G}_2$ $(X_i = xP_i,\ i \in \{1, 2\})$:

- The signing oracle is *useless*:
  - Gives $\sigma \in \mathbb{G}_1$ s.t. $e(\sigma, P_2) = e(H', X)$ for $H' \in_R \mathbb{G}_1$.
  - *But* $\mathcal{A}$ can generate such (H', σ) pair itself – select $y \in_R \mathbb{Z}_n^*$ and compute $H' = yP_1$ and $\sigma = yX_1$.
- So $\mathcal{A}$'s problem is to compute $\sigma = xH$, given $H, X_1 \in \mathbb{G}_1$ and $X_2 \in G_2$.
  - This is precisely co-DHP* in $(\mathbb{G}_1, \mathbb{G}_2)$

BLS-2 or BLS-3?

- Boneh-Lynn-Shacham asserts:
  - co-DHP* is a "stronger complexity assumption" than co-DHP.
  - And hence there is no reason to use BLS-3.
- But BLS-3 is more efficient as it is implemented in the Type 3 setting.

A Concrete Situation

- Let $E/\mathbb{F}_p : Y^2 = X^3 + b$ where $p = 36z^4 + 36z^3 + 24z^2 + 6z + 1$ for some $z \in \mathbb{Z}$.
    - (E can also be viewed as an elliptic curve over any extension field of $\mathbb{F}_p$.)
    - $\#E(\mathbb{F}_p) = n$, where n $(= 36z^4 + 36z^3 + 18z^2 + 6z + 1)$ is a prime.
    - The embedding (*MOV/FR*) degree is $k = 12$.
- $E[n]$: Set of all order-n points P on E.
    - $|E[n]| = n^2$.
    - $E[n] \subseteq E(\mathbb{F}_{p^{12}})$.
- Introduced by Barreto-Naehrig (2005).
    - Ideal for the 128-bit security level.
    - p and n are 256-bit primes.

Type 3 Pairing from BN Curves

- $\mathbb{G}_T$: order-n subgroup of $\mathbb{F}_{p^{12}}$.
- $\mathbb{G}_1 = E(\mathbb{F}_p)$.
- $\mathbb{G}_2$: any other order-n subgroup of $E[n]$.
    - There are n such subgroups of $E[n]$.
- To speedup computation, one sets $\mathbb{G}_2$ to be the "Trace-0" subgroup of $E[n]$.
    - Call it $\mathbb{T}_0$.
    - Elements of $\mathbb{T}_0$ are (essentially) defined over $\mathbb{F}_{p^2}$ (instead of $\mathbb{F}_{p^{12}}$).
- One of the fastest pairings is R-ate pairing $e_3 : \mathbb{G}_1 \times \mathbb{T}_0 \rightarrow \mathbb{G}_T$.
    - Type 3 pairing: no efficiently computable isomorphism $\psi : \mathbb{T}_0 \rightarrow \mathbb{G}_1$ is known.

Type 2 Pairing

- Let $P_2' \in E[n]$ such that $P_2' \notin \mathbb{G}_1 \cup \mathbb{T}_0$.
- Define $\mathbb{G}_2' = \langle P_2' \rangle$.
- Let $e_2 : \mathbb{G}_1 \times \mathbb{G}_2' \to \mathbb{G}_T$.
    - Type 2 pairing: there is an efficiently computable isomorphism from $\mathbb{G}_2'$ to $\mathbb{G}_1$.
- The task of computing e_2 can be reduced to the task of computing e_3 (with a little extra cost).

Type 2 vs. Type 3 Pairings

Why bother about Type 2 or Type 3?

$$\boxed{e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T}$$

- **Functionality:** Some pairing-based protocols use the map ψ in the protocol itself.
- **Security:** Many protocols use the map ψ in their security proofs.
- **Efficiency:** Elements of $\mathbb{G}_2'$ are defined over $\mathbb{F}_{p^{12}}$ and elements of $\mathbb{T}_0$ are defined over $\mathbb{F}_{p^2}$.
  - **Computation:** Arithmetic in $\mathbb{G}_2'$ is roughly 15 times as expensive as in $\mathbb{T}_0$.
  - **Bandwidth:** Elements of $\mathbb{G}_2'$ are 6 times larger than elements of $\mathbb{T}_0$.

Existing *Status*

- Over all Type 3 is a better choice in terms of efficiency.
- Protocols are usually first described in symmetric setting (with a remark: can be instantiated in asymmetric setting).
- In asymmetric setting designers prefer Type 2 pairings instead of Type 3.
    - Perhaps due to the assertion [BLS:2001] that co-DHP* is a stronger complexity assumption than co-DHP.
    - Or assuming that the desired functionality cannot be achieved in Type 3.

Common belief

- [Pairings for cryptographers, 2008]: There exist many primitives in pairing-based cryptography whose security proof does not apply if the cryptosystem is implemented using a Type 3 pairing.
- Some researchers even assumed an oracle access to ψ for the security proof to go through in Type 3 settings.

Two Questions

- Can we make any efficiency gain in Type 2 settings?
- What exact role ψ plays as far as the question of functionality and security of a protocol is concerned?

Defining $\mathbb{G}_1$, $\mathbb{T}_0$ and $\mathbb{G}_2'$

- Let P_2' be a random point in $E[n] \setminus (\mathbb{G}_1 \cup \mathbb{T}_0)$, set $\mathbb{G}_2' = \langle P_2' \rangle$.
- Define $P_1 = \frac{1}{12}\mathrm{Tr}(P_2')$,
  $\mathrm{Tr}(Z) = \sum_{i=0}^{11} \pi^i(Z)$ where $\pi : (x, y) \to (x^p, y^p)$.
- $\psi : \mathbb{G}_2' \to \mathbb{G}_1$, $Q \to \frac{1}{12}\mathrm{Tr}(Q)$ is an efficiently computable isomorphism.
- Set $P_2 = c^{-1}(P_2' - P_1)$ for some arbitrary $c \in \mathbb{Z}_n^*$.
- $P_2 \in \mathbb{T}_0$ and the mapping

$$\rho : \mathbb{G}_2' \to \mathbb{T}_0, \quad Q \to Q - \psi(Q)$$

is an efficiently-computable isomorphism with $\rho(P_2') = cP_2$.

New Representation for $\mathbb{G}_2'$

- Given any $Q \in \mathbb{G}_2'$, one can efficiently compute $Q_1 = \psi(Q)$ and $Q_2 = \rho(Q) = Q - Q_1$; so that $Q = Q_1 + Q_2$.
- Write $D(Q) = (\psi(Q), \rho(Q))$ and let $\mathbb{H}_2' \subseteq \mathbb{G}_1 \times \mathbb{T}_0$ denote the range of D.
  - $D : \mathbb{G}_2' \to \mathbb{H}_2'$ is an efficiently-computable isomorphism whose inverse is also efficiently computable.
- Without loss of generality, any $Q \in \mathbb{G}_2'$ can be represented by a pair of points $(Q_1, Q_2) \in \mathbb{G}_1 \times \mathbb{T}_0$.
  - Arithmetic in $\mathbb{G}_2'$ with this representation is component-wise.

New Representation for $\mathbb{G}_2'$

- Given any $Q \in \mathbb{G}_2'$, one can efficiently compute $Q_1 = \psi(Q)$ and $Q_2 = \rho(Q) = Q - Q_1$; so that $Q = Q_1 + Q_2$.
- Write $D(Q) = (\psi(Q), \rho(Q))$ and let $\mathbb{H}_2' \subseteq \mathbb{G}_1 \times \mathbb{T}_0$ denote the range of D.
  - $D : \mathbb{G}_2' \to \mathbb{H}_2'$ is an efficiently-computable isomorphism whose inverse is also efficiently computable.
- Without loss of generality, any $Q \in \mathbb{G}_2'$ can be represented by a pair of points $(Q_1, Q_2) \in \mathbb{G}_1 \times \mathbb{T}_0$.
  - Arithmetic in $\mathbb{G}_2'$ with this representation is component-wise.

Efficiency: Arithmetic in $\mathbb{G}_2'$ is now only $4/3$ times as expensive as in $\mathbb{T}_0$ – was roughly 15 times.
Bandwidth: $\mathbb{G}_2'$ elements are 1.5 times larger than $\mathbb{T}_0$ elements – was 6 times.

Type 2 versus Type 3: DHP

- We have $\mathbb{G}_2' = \langle P_2' \rangle$, $P_2 = c^{-1}\rho(P_2')$ and $P_1 = \frac{1}{12}\mathrm{Tr}(P_2')$.
- If c is *known* then co-DHP and co-DHP* are *provably* equivalent.
- co-DHP $\leq$ co-DHP*:
  - Given (Q, zP_2'), compute $c^{-1}\rho(zP_2') = zP_2$ and $zP_2' - czP_2 = zP_1$.
  - Use a co-DHP* solver to find the solution zQ of the co-DHP* instance (Q, zP_1, zP_2).
- co-DHP* $\leq$ co-DHP: Similar argument.
- If c is unknown, then co-DHP and co-DHP* *appear* to be unrelated.

Type 2 versus Type 3: BLS

- BLS-3 does not require a stronger complexity assumption as claimed by the designers.
- With the new representation of $\mathbb{G}_2'$, the only difference between BLS-2 and BLS-3 is that the public key X in BLS-2 is slightly larger as it includes an extra $\mathbb{G}_1$ component.
- This $\mathbb{G}_1$ component does not play any role in signature generation and verification.
  - Kind of an appendage – only role is to compute the map ψ.
  - [New paradigms in signature schemes: Shacham, 2005] "the map [ψ] isn't merely a proof artifact".

Type 2 versus Type 3: BLS

- BLS-3 does not require a stronger complexity assumption as claimed by the designers.
- With the new representation of $\mathbb{G}_2'$, the only difference between BLS-2 and BLS-3 is that the public key X in BLS-2 is slightly larger as it includes an extra $\mathbb{G}_1$ component.
- This $\mathbb{G}_1$ component does not play any role in signature generation and verification.
    - Kind of an appendage – only role is to compute the map ψ.
    - [New paradigms in signature schemes: Shacham, 2005] "the map [ψ] isn't merely a proof artifact".
- ψ **does not** play *any* cryptographically significant role as far as the BLS signature is concerned.

BasicIdent in Type 2

$e : \mathbb{G}_1 \times \mathbb{G}_2' \to \mathbb{G}_T$, $\mathbb{G}_1 = \langle P_1 \rangle$ and $\mathbb{G}_2' = \langle P_2' \rangle$.
$H : \{0,1\}^* \to \mathbb{G}_1^*$ is a hash function.

- Set-Up: msk: $s \in_R \mathbb{Z}_p^*$ and set $P_{\text{pub}} = sP_2'$.
- Key-Gen: Given ID $\in \{0,1\}^*$, compute $Q_{\text{ID}} = H(\text{ID})$ and set $d_{\text{ID}} = sQ_{\text{ID}}$.
- Encrypt: To encrypt $M \in \mathbb{G}_T$ to ID compute $Q_{\text{ID}} = H(\text{ID})$, choose $r \in_R \mathbb{Z}_p^*$ and set:

$$C = \langle rP_2', M \times e(Q_{\text{ID}}, P_{\text{pub}})^r \rangle$$

- Decrypt: To decrypt $C = \langle U, V \rangle$ using d_{ID} compute

$$V \times e(d_{\text{ID}}, U)^{-1} = M.$$

Correctness:

$$e(d_{\text{ID}}, U) = e(sQ_{\text{ID}}, rP_2') = e(Q_{\text{ID}}, sP_2')^r = e(Q_{\text{ID}}, P_{\text{pub}})^r.$$

BasicIdent in Type 3

$e : \mathbb{G}_1 \times \mathbb{T}_0 \rightarrow \mathbb{G}_T$, $\mathbb{G}_1 = \langle P \rangle$ and $T_0 = \langle P_2 \rangle$.
$H : \{0,1\}^* \rightarrow \mathbb{T}_0^*$ is a hash function.

- Set-Up: msk: $s \in_R \mathbb{Z}_p^*$ and set $P_{\text{pub}} = sP_1$.
- Key-Gen: Given ID $\in \{0,1\}^*$, compute $Q_{\text{ID}} = H(\text{ID})$ and set $d_{\text{ID}} = sQ_{\text{ID}}$.
- Encrypt: To encrypt $M \in \mathbb{G}_T$ to ID compute $Q_{\text{ID}} = H(\text{ID})$, choose $r \in_R \mathbb{Z}_p^*$ and set:

$$C = \langle rP_1, M \times e(P_{\text{pub}}, Q_{\text{ID}})^r \rangle$$

- Decrypt: To decrypt $C = \langle U, V \rangle$ using d_{ID} compute

$$V \times e(U, d_{\text{ID}})^{-1} = M.$$

Correctness:

$$e(U, d_{\text{ID}}) = e(rP_1, sQ_{\text{ID}}) = e(sP_1, Q_{\text{ID}})^r = e(P_{\text{pub}}, Q_{\text{ID}})^r.$$

BF-IBE in asymmetric setting

- Can be instantiated in both Type 2 and Type 3.
    - Type 3 allows more efficient implementation.
    - The underlying complexity assumption was thought to be "somewhat unnatural".
    - So Type 2 settings was suggested for implementation.
- (D)BDH in Type 3 is at least as hard as in Type 2.
    - Type 3 is overall a better choice for BF-IBE.

Type 2 versus Type 3

More generally...

- Given *any* protocol, Protocol-2, described in Type 2 setting, and a security argument for Protocol-2 *wrt* some hard problem $\mathcal{P}$-2, there is:
  - A *natural* transformation of Protocol-2 to Protocol-3 that uses a Type 3 pairing.
  - A *natural* transformation of $\mathcal{P}$-2 to $\mathcal{P}$-3.
  - A *natural* transformation of the security argument to one for Protocol-3 wrt $\mathcal{P}$-3.
- Protocol-3 is usually more efficient than Protocol-2.
- $\mathcal{P}$-3 is at least as hard as $\mathcal{P}$-2 (for appropriately chosen parameters).

Role of ψ revisited

- It appears ψ **does not** play any significant role in terms of
  - functionality
  - security.
- Use of ψ may have a negative impact on efficiency.
- *No* reason to use Type 2 instead of Type 3 setting.

Role of ψ revisited

- It appears ψ **does not** play any significant role in terms of
    - functionality
    - security.
- Use of ψ may have a negative impact on efficiency.
- *No* reason to use Type 2 instead of Type 3 setting.
    - *But we no longer have the scope to use one more Greek symbol in our papers!*

Type 4 Pairings

Type 4 Pairing

$$\boxed{e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T}$$

- $\mathbb{G}_1$ and $\mathbb{G}_T$ are cyclic groups of prime order n.
- *But* $\mathbb{G}_2$ is a non-cyclic group of order n^2.
- One can efficiently compute the homomorphism $\psi : \mathbb{G}_2 \to \mathbb{G}_1$.
- One can hash onto $\mathbb{G}_2$.

Why Type 4?

Consider the following ID-based Key Agreement Protocol [Scott]:

- Public hash function $H : \{0,1\}^* \to \mathbb{G}_2$.
- PKG has a master secret $s \in_R [1, n-1]$.
- User A's pub. key is $Q_A = H(ID_A)$ and PKG provides pvt. key $d_A = sQ_A$.
- Similarly, B's pub. key is $Q_B = H(ID_B)$ and pvt. key $d_B = sQ_B$.
- A and B exchange some messages and compute a shared secret

$$K = e(\psi(Q_A), Q_B)^{sxy}.$$

Type 4 Pairing from BN Curves

- Let $\mathbb{G}_1 = E(\mathbb{F}_p)$ and $\mathbb{G}_T$ be the order-n subgroup of $\mathbb{F}^*_{p^{12}}$.
- $e_4 : \mathbb{G}_1 \times E[n] \to \mathbb{G}_T$ is a Type 4 pairing.
- Bitlength of (compressed) elements of $\mathbb{G}_1$: 257 and of $E[n]$: 3073.
- $\mathrm{Tr}(P) = \sum_{i=0}^{11} \pi^i(P)$ is an efficiently computable homomorphism from $E[n]$ to $\mathbb{G}_1$.
    - π is the p-th power Frobenius.
- Hashing onto $E[n]$ is possible *but* costly.
    - Roughly 540 times the cost of a point multiplication in $\mathbb{G}_1$ [Chen-Cheng-Smart:2007].
    - Type 3 Pairing is around 10-times the cost of a point multiplication.
    - Hashing onto $\mathbb{G}_1$ is *very* cheap.

The Trace-0 Subgroup *(Recap.)*

- $E[n]$ contains $n+1$ subgroups of order n.
- One of them is $\mathbb{G}_1$.
- Another is the "Trace-0" subgroup, $\mathbb{T}_0$.
    - All points $P \in E[n]$ satisfying $\text{Tr}(P) = \sum_{i=0}^{11} \pi^i(P) = \infty$.
- $\mathbb{T}_0$ can be viewed as having coordinates in $\mathbb{F}_{p^2}$ (instead of $\mathbb{F}_{p^{12}}$).
- Recall that we defined $e_3 : \mathbb{G}_1 \times \mathbb{T}_0 \to \mathbb{G}_T$.

Efficient Representation of $E[n]$

- Define $\psi : E[n] \rightarrow \mathbb{G}_1$ by $\psi(Q) = \frac{1}{12}\mathrm{Tr}(Q)$.
  - $Q - \psi(Q) \in \mathbb{T}_0$ for all $Q \in E[n]$.
  - $\rho : Q \mapsto Q - \psi(Q)$ is a homomorphism from $E[n]$ onto $\mathbb{T}_0$.
- Define $\phi : E[n] \rightarrow \mathbb{G}_1 \times \mathbb{T}_0$ by $\phi(Q) = (\psi(Q), \rho(Q))$.
  - ϕ is an efficiently-computable isomorphism,
  - The inverse mapping, given by $(Q_1, Q_2) \mapsto Q_1 + Q_2$, is also efficiently computable.
- Wlg, elements of $E[n]$ can be represented as pairs of points $(Q_1, Q_2) \in \mathbb{G}_1 \times \mathbb{T}_0$.

Hashing onto $E[n]$

- Define $H : \{0,1\}^* \to E[n]$ as $H(m) = (H_1(m), H_2(m))$.
  - $H_1 : \{0,1\}^* \to \mathbb{G}_1$.
  - $H_2 : \{0,1\}^* \to \mathbb{T}_0$.
- Significantly faster hashing onto $E[n]$.
  - Hashing onto $\mathbb{G}_1$ and $\mathbb{T}_0$ requires arithmetic in $\mathbb{F}_p$ and $\mathbb{F}_{p^2}$.
  - Less than 3 times as costly as a point multiplication in $\mathbb{G}_1$.
- Hashing into $E[n]$ is 180 times less costly than previously estimated.
- If H_1 and H_2 are modeled as random oracles, then H is also a random oracle.

Type 4 Pairing from Type 3

- Let $e_3 : \mathbb{G}_1 \times \mathbb{T}_0 \to \mathbb{G}_T$ be a Type 3 pairing.
- Define $e_4 : \mathbb{G}_1 \times E[n] \to \mathbb{G}_T$ by $e_4(P, Q) = e_3(P, \hat{Q})$, where $\hat{Q} = Q - \pi^6(Q)$.
    - If $Q = (Q_1, Q_2)$, then $\hat{Q} = (\infty, 2Q_2)$.
- e_4 is bilinear and can be computed in essentially the same time as e_3.
- e_4 is non-degenerate:
    1. $\forall P \in \mathbb{G}_1 \setminus \{\infty\}, \exists Q \in E[n]$ s.t. $e_4(P, Q) \neq 1$.
    2. $\forall Q \in E[n] \setminus \mathbb{G}_1, \exists P \in \mathbb{G}_1$ s.t. $e_4(P, Q) \neq 1$.

Comparison

$E/\mathbb{F}_p : Y^2 = X^2 + 3$ with BN parameters $z = \texttt{6000000000001F2D}$.

	Type 3	Type 4
Bitlength of elements in $\mathbb{G}_1$	257	257
Bitlength of elements in $\mathbb{T}_0/E[n]$	513	770
Bitlength of elements in $\mathbb{G}_T$	1,024	1,024
Exponentiation in $\mathbb{G}_1$	1,533m	1,533m
Exponentiation in $\mathbb{T}_0/E[n]$	3,052m	4,585m
Hashing into $\mathbb{G}_1$	315m	315m
Hashing into $\mathbb{T}_0/E[n]$	3,726m	4,041m
Pairing	15,175m	15,175m

(Estimated costs are in terms of $\mathbb{F}_p$ multiplications.)

Protocols in Type 4

- Consider Type 4 setting only when
  - The protocol requires hashing into $\mathbb{G}_2$.
  - Then apply ψ on the hash output.
- Remember $\mathbb{G}_2 = E[n]$ has order n^2.
  - May affect functionality and security in a critical way.

Group Signature

- Every member has a secret key but there is a single public key for the whole group.
- Group signatures provide signer-anonymity.
- Revocation of a user may be critical for some applications.

Boneh-Shacham Group Signature

- BS group signature allows a verifier to locally check whether the given signature is generated by a revoked user.
    - Verifier-local revocation (VLR) group signature.
    - The signature length is *short*.
    - Application: privacy preserving attestation.
- Can be implemented in Type 1 but *not* in Type 2 or Type 3.
- The first protocol for which Type 4 setting was introduced [Shacham:2005].

The BS-VLR Protocol

Some essentials:

- Employs a Type 4 pairing $e_4 : \mathbb{G}_1 \times E[n] \to \mathbb{G}_T$.
- Two hash functions (treated as random oracle):
  - $H_0 : \{0,1\}^* \to E[n] \times E[n]$.
  - $H : \{0,1\}^* \to [1, n-1]$.
- Group public key, $gpk = (P_1, P_2, W)$ where $P_2 \in_R E[n]$, $P_1 = \psi(P_2)$ and $W = \gamma P_2$, $\gamma \in_R [1, n-1]$.
- Pvt. key of user i, $gsk[i] = (A_i, x_i)$, where $x_i \in_R [1, n-1]$ and $A_i = (\gamma + x_i)^{-1} P_1$.
  - The corresponding revocation token is A_i.

Signing and Verification

The protocol is quite involved.

- To sign i computes:
  - $(\hat{U}, \hat{V}) = H_0(gpk, M, r)$ where M is the message and $r \in_R [1, n-1]$.
  - $U = \psi(\hat{U})$ and $V = \psi(\hat{V})$.
  - $T_1 = \alpha U$ and $T_2 = A_i + \alpha V$, where $\alpha \in_R [1, n-1]$.
  - Compute helper values R_1, R_2, R_3.
  - Challenge value, $c = H(gpk, M, r, T_1, T_2, R_1, R_2, R_3)$.
  - σ contains r, T_1, T_2, c and three randomizers (to rederive R_1, R_2, R_3).
- σ is accepted as valid if c is a correct challenge *and* the signer is not revoked.

Revocation Check in BS-VLR Group Signature

- A list of revocation tokens (RL) corresponding to the revoked users is publicly available.
- Suppose the signature (σ) is generated by a user i whose revocation token A_i is in RL.
- The correctness of the protocol mandates that σ must be rejected.

Revocation Check

- The protocol stipulates that σ will be rejected since:

$$e_4(T_2 - A_i, \hat{U}) = e_4(T_1, \hat{V}) \tag{1}$$

  - $(\hat{U}, \hat{V}) = H_0(gpk, M, r) \in E[n] \times E[n]$.
  - $T_1 = \alpha U$ and $T_2 = A_i + \alpha V$, where $U = \psi(\hat{U})$ and $V = \psi(\hat{V})$.
- So Eqn. 1 can be rewritten as:

$$e_4(\alpha V, \hat{U}) = e_4(\alpha U, \hat{V}) \tag{2}$$

- Trivially holds *if* both $\hat{U}$, $\hat{V}$ are from same order-n subgroup of $E[n]$.
  - Write $\hat{U} = x\hat{V}$ (and $U = xV$).

Another Look at the Revocation Check

- But $E[n]$ is a group of order n^2!
- $\hat{U}, \hat{V}$ are obtained through hashing into random elements of $E[n]$.
  - The probability that they belong to the same order-n subgroup of $E[n]$ is negligibly small.
- With overwhelming probability Eqn. 2 **will not** hold.
  - A signature generated by a revoked user will be accepted as valid.

Another Look at the Revocation Check

- But $E[n]$ is a group of order n^2!
- $\hat{U}, \hat{V}$ are obtained through hashing into random elements of $E[n]$.
  - The probability that they belong to the same order-n subgroup of $E[n]$ is negligibly small.
- With overwhelming probability Eqn. 2 **will not** hold.
  - A signature generated by a revoked user will be accepted as valid.
- The protocol is **not** secure!
  - So also the protocols that extend the idea of BS-VLR group signature.
    - Nakanishi and Funabiki [2006].
    - Bringer et al. [2008].

What's Wrong with the Security Argument?

- Security mandates that the protocol satisfies correctness, traceability and selfless-anonymity properties.
- Fails to satisfy the correctness property when we work in $E[n]$.
  - By implication the traceability property is violated.
- The arguments hold *if* we restrict to a order-n subgroup of $E[n]$.
  - Not instantiable in Type 2 or Type 3 settings.
  - Can be instantiated in Type 1, *but* the signature is no longer short.

Rescuing BS-VLR Scheme

Essential idea:

- For random $\hat{U}, \hat{V} \in E[n]$ in general one cannot expect

$$e\left(\alpha\psi(\hat{V}), \hat{U}\right) = e\left(\alpha\psi(\hat{U}), \hat{V}\right).$$

- But *bilinearity* of e ensures

$$e\left(\alpha\psi(\hat{V}), \hat{U}\right) = e\left(\psi(\hat{V}), \alpha\hat{U}\right).$$

- Revocation check works properly if we send $\hat{T}_1 = \alpha\hat{U} \in E[n]$ instead of $T_1 \in \mathbb{G}_1$ as part of σ.
  - For each $A \in \mathrm{RL}$ check whether the following holds:

$$e(T_2 - A, \hat{U}) = e(V, \hat{T}_1).$$

Modified BS-VLR

- Key generation algorithm remains unchanged.
- Define $H_0 : \{0,1\}^* \to E[n] \times \mathbb{G}_1$.
- To sign, compute $(\hat{U}, V) = H_0(gpk, M, r)$ and $\hat{T}_1 = \alpha\hat{U}$.
  - Use $\hat{T}_1$ and $\hat{U}$ to compute R_3 and $\hat{T}_1$ to compute c.
  - Send $\hat{T}_1 \in E[n]$ (*not* $T_1 \in \mathbb{G}_1$) as part of σ.
- To verify:
  - Use $\hat{T}_1$, $\hat{U}$ to rederive R_3.
  - Use $T_1 = \psi(\hat{T}_1)$ to rederive R_1.
  - Use $\hat{T}_1$ to rederive c.
  - Use $\hat{T}_1$ in revocation check.
- The signature now contains an element of $E[n]$.
  - With the new representation, the increase in signature length is only 513 bits.

Is the Protocol Secure?

It appears so!

- Correctness is easy to check.
- The original argument concerning traceability should also carry over.
- But the selfless-anonymity argument requires some twists!
  - The original argument is flawed.
  - Concerns a proper simulation of H_0.
  - Must ensure that each query to H_0 returns a random element from $E[n]$.
- The new representation of $E[n]$ comes to the rescue.

Selfless-Anonymity

Essential idea:

- Decision Linear problem:
  Given $(U_0, U_1, aU_0, bU_1, V, Z)$ where $U_0, U_1 \in_R E[n]$,
  $a, b \in_R [1, n-1]$, decide whether $Z = (a+b)V$ or $Z \in_R \mathbb{G}_1$.
- In the original argument, the RO output $\hat{U}$ is restricted to the order-n subgroup $\langle U_0 \rangle$.
  - For a proper simulation $\hat{U}$ must be a random element of $E[n]$.
- Represent $U_0 = (U_{0,1}, U_{0,2}) \in \mathbb{G}_1 \times \mathbb{T}_0$.
- From U_0 one can derive another random element in $E[n]$ as $\tilde{U}_0 = (U_{0,1}, xU_{0,2})$ where $x \in_R [1, n-1]$.
- Allows a proper simulation of the RO and also to patch the signature component $\hat{T}_1$ appropriately.

Protocols w/o Random Oracle

Identity-Based Encryption

- Waters 2005:
  - Security is based on DBDH.
  - Rather large public parameters.
  - Reduction is *not* tight.
- Gentry 2006:
  - Simple construction with constant size public parameter.
  - Tight security reduction...*but* based on non-static, non-standard assumption (decisional q-ABDHE).

Dual system encryption

[Waters 2009]

- Constant size public parameters.
- Security based on static assumptions (DBDH and DLIN).
- Reduction is not tight.
- Polynomial degradation for hierarchical IBE.

A variant of Waters IBE

$$e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T,\ \mathbb{G} = \langle P \rangle$$

ID: ℓ blocks of k/ℓ-bits.

- **Set-Up:** Choose random $x \in \mathbb{Z}_p^*$ and $P, P_2, U' U_1, \ldots, U_n$ from $\mathbb{G}$. Set $xP = P_1$.
  msk: xP_2 and PP $= (P, P_1, P_2, U', U_1, \ldots, U_\ell)$.
- **Key-Gen:** ID $= (\mathrm{ID}_1, \ldots, \mathrm{ID}_\ell)$, ID_i is a block of k/ℓ-bits.

$$d_{\mathrm{ID}} = (xP_2 + rV, rP) \qquad \text{where}$$

$$V = U' + \sum_{i=1}^{\ell} \mathrm{ID}_i U_i.$$

Waters IBE (Contd.)

$$\boxed{V(\mathsf{ID}) = U' + \textstyle\sum_{i=1}^{\ell} \mathsf{id}_i U_i; \quad \mathsf{ID} = (\mathsf{id}_1, \ldots, \mathsf{id}_\ell}$$

- **Encrypt:** $M \in \mathbb{G}_T$ encrypted for ID as

$$C = (e(P_1, P_2)^t \times M, tP, tV).$$

- **Decrypt:** Decrypt $C = (C_1, C_2, C_3)$ using $d_{\mathsf{ID}} = (d_1, d_2)$ as

$$C_1 \times \frac{e(d_2, C_3)}{e(d_1, C_2)}.$$

Correctness:

$$\frac{e(d_2, C_3)}{e(d_1, C_2)} = \frac{e(rP, tV)}{e(xP_2 + rV, tP)} = \frac{1}{e(P_1, P_2)^t}$$

Signature

- Boneh-Boyen *short* signature [2004].
- Waters signature [2005].
  - Obtained through Naor's transformation on Waters IBE.
  - Security is based on CDH.

Boneh-Boyen Signature

- **Key-Gen:** $P_1 \in_R \mathbb{G}_1$ and $P_2 \in_R \mathbb{G}_2$, $x, y \in_R \mathbb{Z}_n^*$, compute $U = xP_2$, $V = yP_2$ and $Z = e(P_1, P_2)$. Pub. key: (P_1, P_2, U, V, Z) and sec. key: (x, y).
- **Signing:** Given sk: (x, y) and mesg. $m \in \mathbb{Z}_n^*$; pick $r \in_R \mathbb{Z}_n^*$, compute $\sigma = 1/(x + m + yr)P_1$. The signature is (σ, r).
- **Verication:** Given pk $= (P_1, P_2, U, V, Z)$, $m \in \mathbb{Z}_n^*$ and (σ, r), accept iff
$$e(\sigma, U + mP_2 + rV) = Z.$$

Correctness:
$e(\sigma, U + mP_2 + rV) = e(1/(x + m + yr)P_1, (x + m + ry)P_2) = e(P_1, P_2)$.
Security is based on a non-static assumption: strong q-DH.
The question of equivalence...

Pairing over composite order groups

$$\boxed{e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T}$$

$|\mathbb{G}| = |\mathbb{G}_T| = pq$, (p, q): secret (prime).

- Introduced by Boneh-Goh-Nissim [2005].
  - Homomorphic encryption.
  - Private Information Retrieval.
- Non-interactive zero knowledge [Groth-Sahai 2008].
- Hierarchical IBE [Lewko-Waters 2010].
- Traitor tracing, attribute-based encryption...

Composite order setting

- Subgroup decision assumption:
    - Given $P \in \mathbb{G}$ decide whether P is of order p.
    - $n(= pq)$ must be infeasible to factor.
- [Freeman:2010] A framework to convert protocols from composite to prime order groups.
    - [Meiklejohn-Shacham-Freeman:2010] Not all protocols can be so converted.
- [Boneh-Rubin-Silverberg:2009] Composite order pairing-friendly groups.
- [Kobliz:2010] "A Security Weakness in Composite-Order Pairing-Based Protocols with Imbedding Degree $k > 2$."

Research directions

- Protocols
  - Design new protocols, e.g., employ the dual-system encryption paradigm.
  - Improve existing ones.
- Introduce a new pairing or pairing-friendly curves.
- Efficient implementation of pairing (and protocols).
  - From 16 min. to less than a millisecond...
- Analyse pairing-based protocols in terms of *functionality, security and efficiency*.
- Move to higher genus.
  - Some work has appeared for Genus-2 pairing.

Still interested?

Still interested?

- Rush to Yamanaka Hot Spring to attend Pairing 2010.

Still interested?

- Rush to Yamanaka Hot Spring to attend Pairing 2010.
- Talk to people at the sideline of Indocrypt 2010.

Still interested?

- Rush to Yamanaka Hot Spring to attend Pairing 2010.
- Talk to people at the sideline of Indocrypt 2010.
- Sit back and relax...
  - I guess there will be another tutorial on Pairing-Based Crypto at Indocrypt 2018.

In *lieu* of a conclusion

Where the mind is without fear and the head is held high
Where knowledge is free
Where the world has not been broken up into fragments
By narrow domestic walls
Where words come out from the depthsof truth
Where tireless striving stretches its arms towards perfection
Where the clear stream of reason has not lost its way
Into the dreary desert sand of dead habit
Where the mind is led forward by thee
Into ever-widening thought and action
Into that heaven of freedom, my Father, let my country awake.