Converting Pairing-Based Cryptosystems from Composite-Order Groups to Prime-Order Groups

David Mandell Freeman

Stanford University, USA

Eurocrypt 2010
Monaco, Monaco

31 May 2010

Composite-order bilinear groups: What are they?

- Cyclic groups $\mathbb{G}, \mathbb{G}_t$ of order $N = p_1 \cdots p_r$;
- Nondegenerate, bilinear pairing $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_t$;
- Useful for crypto if (some version of) the *subgroup decision assumption* holds in $\mathbb{G}$:

$$\{x \xleftarrow{\mathrm{R}} \mathbb{G} : \mathrm{ord}(x) < N\} \quad \text{and} \quad \{x \xleftarrow{\mathrm{R}} \mathbb{G}\}$$

  computationally indistinguishable.
- In particular, factoring N must be infeasible.

Composite-order bilinear groups: What are they good for?

Used in recent years to solve many cryptographic problems:

- "Somewhat homomorphic" encryption [BGN05]
- Traitor tracing [BSW06]
- Ring and group signatures [BW07, SW07]
- NIZK proof systems [GOS06, GS08]
- Attribute-based encryption [KSW08, LOSTW10]
- Fully secure HIBE [W09, LW10]

Composite-order bilinear groups: Some drawbacks

Groups are instantiated using supersingular elliptic curves E over finite fields $\mathbb{F}_q$, $q \equiv -1 \pmod{N}$ prime.

- Groups are very large: $N \approx 2^{1024}$ to prevent factoring attack.
- Pairings are **very** slow [Scott].

usual pairing-based crypto: (prime-order MNT curve)	$\mathbb{G} \subset E(\mathbb{F}_q) \sim$ **160 bits** $\mathbb{G}_t \subset \mathbb{F}^*_{q^6} \sim$ **1024 bits** $\sim$ **3 ms** pairing
composite-order groups: (supersingular curve)	$\mathbb{G} \subset E(\mathbb{F}_q) \sim$ **1024 bits** $\mathbb{G}_t \subset \mathbb{F}^*_{q^2} \sim$ **2048 bits** $\sim$ **150 ms** pairing

Conclusion: using composite-order elliptic curves negates many advantages of elliptic curve crypto.

Our goal:

Obtain *functionality* of composite-order group cryptosystems using *infrastructure* of prime-order bilinear groups:

small group sizes

fast pairing

well studied assumptions

- Want a general conversion method.
- Previous solutions [IP08, W09, LW10] ad-hoc (or at least opaque).

Our contribution

- Abstract framework that captures the cryptographic properties of composite-order bilinear groups.
- Instantiations of groups with these properties using prime-order bilinear groups.
- Method for converting cryptosystems from composite-order groups to prime-order groups.
  - Not a black-box compiler; proofs need to be checked (fails for [LW10]).
- Conversion of
  1. "Somewhat homomorphic" encryption [BGN05];
  2. Traitor tracing [BSW06];
  3. Attribute-based encryption [KSW08].

Generalizing the subgroup decision assumption

Generalized subgroup decision problem:

- 5 groups $G_1 \subset G,\ H_1 \subset H,\ G_t$
- nondegenerate bilinear map $e \colon G \times H \to G_t$ (asymmetric)
- distinguish $\{x \xleftarrow{\mathrm{R}} G_1\}$ from $\{x \xleftarrow{\mathrm{R}} G\}$

  or

  distinguish $\{y \xleftarrow{\mathrm{R}} H_1\}$ from $\{y \xleftarrow{\mathrm{R}} H\}$.

If both problems computationally infeasible, then *generalized subgroup decision assumption* holds for (G, G_1, H, H_1, G_t, e).

A key observation [CS03, G04]

DDH is a subgroup decision problem!

- Given group $\mathbb{G}_1$ of order p, define $G := \mathbb{G}_1 \times \mathbb{G}_1$.
- $G_1 :=$ random linear subgroup $\langle (g, g^x) \rangle$.
- Then $(g^y, g^z) \in G_1 \Leftrightarrow z = xy \pmod{p}$.

Extend to the (asymmetric) pairing setting:

- If $\hat{e}\colon \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_t$ is a pairing, define $H := \mathbb{G}_2 \times \mathbb{G}_2$.
- $H_1 :=$ random linear subgroup $\langle (h, h^{x'}) \rangle$.
- Define $e\colon G \times H \to G_t = \mathbb{G}_t$ by

$$e((g, g'), (h, h')) := \hat{e}(g, h)^a \hat{e}(g, h')^b \hat{e}(g', h)^c \hat{e}(g', h')^d.$$

- Can define pairing into $G_t = \mathbb{G}_t^m$ componentwise.

Theorem

If DDH assumption holds in $\mathbb{G}_1$ and $\mathbb{G}_2$, then generalized subgroup decision assumption holds for (G, G_1, H, H_1, G_t, e).

A key observation [CS03, G04]

DDH is a subgroup decision problem!

- Given group $\mathbb{G}_1$ of order p, define $G := \mathbb{G}_1 \times \mathbb{G}_1$.
- $G_1 :=$ random linear subgroup $\langle (g, g^x) \rangle$.
- Then $(g^y, g^z) \in G_1 \Leftrightarrow z = xy \pmod{p}$.

Extend to the (asymmetric) pairing setting:

- If $\hat{e}\colon \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_t$ is a pairing, define $H := \mathbb{G}_2 \times \mathbb{G}_2$.
- $H_1 :=$ random linear subgroup $\langle (h, h^{x'}) \rangle$.
- Define $e\colon G \times H \to G_t = \mathbb{G}_t$ by

$$e((g, g'), (h, h')) := \hat{e}(g, h)^a \hat{e}(g, h')^b \hat{e}(g', h)^c \hat{e}(g', h')^d.$$

- Can define pairing into $G_t = \mathbb{G}_t^m$ componentwise.

Theorem

If DDH assumption holds in $\mathbb{G}_1$ and $\mathbb{G}_2$, then generalized subgroup decision assumption holds for (G, G_1, H, H_1, G_t, e).

A key observation [CS03, G04]

DDH is a subgroup decision problem!

- Given group $\mathbb{G}_1$ of order p, define $G := \mathbb{G}_1 \times \mathbb{G}_1$.
- $G_1 :=$ random linear subgroup $\langle (g, g^x) \rangle$.
- Then $(g^y, g^z) \in G_1 \Leftrightarrow z = xy \pmod{p}$.

Extend to the (asymmetric) pairing setting:

- If $\hat{e} \colon \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_t$ is a pairing, define $H := \mathbb{G}_2 \times \mathbb{G}_2$.
- $H_1 :=$ random linear subgroup $\langle (h, h^{x'}) \rangle$.
- Define $e \colon G \times H \to G_t = \mathbb{G}_t$ by

$$e((g, g'), (h, h')) := \hat{e}(g, h)^a \hat{e}(g, h')^b \hat{e}(g', h)^c \hat{e}(g', h')^d.$$

- Can define pairing into $G_t = \mathbb{G}_t^m$ componentwise.

Theorem

If DDH assumption holds in $\mathbb{G}_1$ and $\mathbb{G}_2$, then generalized subgroup decision assumption holds for (G, G_1, H, H_1, G_t, e).

But wait...
Isn't DDH easy in groups with a pairing?

1. Not necessarily:
   - DDH believed to be hard on *ordinary* pairing-friendly elliptic curves when $\mathbb{G}_1$ is the *base field subgroup*, $\mathbb{G}_2$ is the *trace-zero subgroup*.
   - Pairing is asymmetric (no efficient maps $\mathbb{G}_1 \leftrightarrow \mathbb{G}_2$).
   - Also called "SXDH" assumption.
2. Yes, if $\mathbb{G}_1 = \mathbb{G}_2$...
   But the *k-linear assumption* may still hold! (with $k \geq 2$)
   - k-linear assumption [HK07, S07] generalizes DDH (*is* DDH when $k = 1$), may hold in groups with k-linear map.
   - Generalize DDH construction: $G = H = \mathbb{G}_1^{k+1}$,
     $G_1 = H_1 =$ random k-dimensional subgroup.
   - k-linear assumption $\Rightarrow$ subgroup decision assumption.

Solution (2) is less efficient: G is larger (more copies of $\mathbb{G}_1$) and not suited to high security levels (bounded embedding degree for symmetric pairings).

But wait...
Isn't DDH easy in groups with a pairing?

1. Not necessarily:
   - DDH believed to be hard on *ordinary* pairing-friendly elliptic curves when $\mathbb{G}_1$ is the *base field subgroup*, $\mathbb{G}_2$ is the *trace-zero subgroup*.
   - Pairing is asymmetric (no efficient maps $\mathbb{G}_1 \leftrightarrow \mathbb{G}_2$).
   - Also called "SXDH" assumption.
2. Yes, if $\mathbb{G}_1 = \mathbb{G}_2$...
   But the *k-linear assumption* may still hold! (with $k \geq 2$)
   - k-linear assumption [HK07, S07] generalizes DDH (*is* DDH when $k = 1$), may hold in groups with k-linear map.
   - Generalize DDH construction: $G = H = \mathbb{G}_1^{k+1}$, $G_1 = H_1 =$ random k-dimensional subgroup.
   - k-linear assumption $\Rightarrow$ subgroup decision assumption.

Solution (2) is less efficient: G is larger (more copies of $\mathbb{G}_1$) and not suited to high security levels (bounded embedding degree for symmetric pairings).

But wait... Isn't DDH easy in groups with a pairing?

1. Not necessarily:
   - DDH believed to be hard on *ordinary* pairing-friendly elliptic curves when $\mathbb{G}_1$ is the *base field subgroup*, $\mathbb{G}_2$ is the *trace-zero subgroup*.
   - Pairing is asymmetric (no efficient maps $\mathbb{G}_1 \leftrightarrow \mathbb{G}_2$).
   - Also called "SXDH" assumption.
2. Yes, if $\mathbb{G}_1 = \mathbb{G}_2$...
   But the *k-linear assumption* may still hold! (with $k \geq 2$)
   - k-linear assumption [HK07, S07] generalizes DDH (*is* DDH when $k = 1$), may hold in groups with k-linear map.
   - Generalize DDH construction: $G = H = \mathbb{G}_1^{k+1}$, $G_1 = H_1 =$ random k-dimensional subgroup.
   - k-linear assumption $\Rightarrow$ subgroup decision assumption.

Solution (2) is less efficient: G is larger (more copies of $\mathbb{G}_1$) and not suited to high security levels (bounded embedding degree for symmetric pairings).

But wait...
Isn't DDH easy in groups with a pairing?

1. Not necessarily:
   - DDH believed to be hard on *ordinary* pairing-friendly elliptic curves when $\mathbb{G}_1$ is the *base field subgroup*, $\mathbb{G}_2$ is the *trace-zero subgroup*.
   - Pairing is asymmetric (no efficient maps $\mathbb{G}_1 \leftrightarrow \mathbb{G}_2$).
   - Also called "SXDH" assumption.
2. Yes, if $\mathbb{G}_1 = \mathbb{G}_2$...
   But the *k-linear assumption* may still hold! (with $k \geq 2$)
   - k-linear assumption [HK07, S07] generalizes DDH (*is* DDH when $k = 1$), may hold in groups with k-linear map.
   - Generalize DDH construction: $G = H = \mathbb{G}_1^{k+1}$,
     $G_1 = H_1 =$ random k-dimensional subgroup.
   - k-linear assumption $\Rightarrow$ subgroup decision assumption.

Solution (2) is less efficient: G is larger (more copies of $\mathbb{G}_1$) and not suited to high security levels (bounded embedding degree for symmetric pairings).

What about the pairing?

Can't use just any pairing e on product groups G and H — cryptosystems require certain properties for correctness.

1. *Projecting* pairing:

$$\begin{array}{rlll} \text{maps:} & \pi_1 \colon G \to G, & \pi_2 \colon H \to H, & \pi_t \colon G_t \to G_t \\ \text{kernels:} & G_1 \subset \ker \pi_1, & H_1 \subset \ker \pi_2, & G_t' \subset \ker \pi_t \\ \text{pairing:} & e(\pi_1(g), \pi_2(h)) = \pi_t(e(g,h)) & & \end{array}$$

2. *Cancelling* pairing:

$$\begin{array}{rll} \text{groups:} & G \cong G_1 \times \cdots \times G_r, & H \cong H_1 \times \cdots \times H_r \\ \text{pairing:} & e(G_i, H_j) = 1 \text{ for } i \neq j. & \end{array}$$

In systems: use G_1 to "blind" elements of G; remove blinding by applying π_1 (projecting) or pairing with elements of H_2 (cancelling).

What about the pairing?

Can't use just any pairing e on product groups G and H — cryptosystems require certain properties for correctness.

1. *Projecting* pairing:

$$\begin{array}{rl} \text{maps:} & \pi_1 : G \to G, \quad \pi_2 : H \to H, \quad \pi_t : G_t \to G_t \\ \text{kernels:} & G_1 \subset \ker \pi_1, \quad H_1 \subset \ker \pi_2, \quad G_t' \subset \ker \pi_t \\ \text{pairing:} & e(\pi_1(g), \pi_2(h)) = \pi_t(e(g,h)) \end{array}$$

2. *Cancelling* pairing:

$$\begin{array}{rl} \text{groups:} & G \cong G_1 \times \cdots \times G_r, \quad H \cong H_1 \times \cdots \times H_r \\ \text{pairing:} & e(G_i, H_j) = 1 \text{ for } i \neq j. \end{array}$$

In systems: use G_1 to "blind" elements of G; remove blinding by applying π_1 (projecting) or pairing with elements of H_2 (cancelling).

What about the pairing?

Can't use just any pairing e on product groups G and H — cryptosystems require certain properties for correctness.

1. *Projecting* pairing:

$$\begin{array}{rl} \text{maps:} & \pi_1 \colon G \to G, \quad \pi_2 \colon H \to H, \quad \pi_t \colon G_t \to G_t \\ \text{kernels:} & G_1 \subset \ker \pi_1, \quad H_1 \subset \ker \pi_2, \quad G_t' \subset \ker \pi_t \\ \text{pairing:} & e(\pi_1(g), \pi_2(h)) = \pi_t(e(g,h)) \end{array}$$

2. *Cancelling* pairing:

$$\begin{array}{rl} \text{groups:} & G \cong G_1 \times \cdots \times G_r, \quad H \cong H_1 \times \cdots \times H_r \\ \text{pairing:} & e(G_i, H_j) = 1 \text{ for } i \neq j. \end{array}$$

In systems: use G_1 to "blind" elements of G; remove blinding by applying π_1 (projecting) or pairing with elements of H_2 (cancelling).

What about the pairing?

Can't use just any pairing e on product groups G and H — cryptosystems require certain properties for correctness.

1. *Projecting* pairing:

$$\begin{array}{rlll} \text{maps:} & \pi_1 \colon G \to G, & \pi_2 \colon H \to H, & \pi_t \colon G_t \to G_t \\ \text{kernels:} & G_1 \subset \ker \pi_1, & H_1 \subset \ker \pi_2, & G_t' \subset \ker \pi_t \\ \text{pairing:} & e(\pi_1(g), \pi_2(h)) = \pi_t(e(g,h)) & & \end{array}$$

2. *Cancelling* pairing:

$$\begin{array}{rll} \text{groups:} & G \cong G_1 \times \cdots \times G_r, & H \cong H_1 \times \cdots \times H_r \\ \text{pairing:} & e(G_i, H_j) = 1 \text{ for } i \neq j. & \end{array}$$

In systems: use G_1 to "blind" elements of G; remove blinding by applying π_1 (projecting) or pairing with elements of H_2 (cancelling).

Projecting and cancelling pairings on product groups

View group elements as vectors $g^{\vec{v}} = (g^{v_1}, g^{v_2})$.
Do linear algebra in the exponent.

1. *Projecting* pairing takes *tensor product* of vectors:
   - Define $e\colon G \times H \to G_t := \mathbb{G}_t^4$ to be vector of all 4 componentwise pairings $\hat{e}$ on $\mathbb{G}_1 \times \mathbb{G}_2$.
   - π_1, π_2, π_t do linear projection in the exponent (details in paper).
2. *Cancelling* pairing takes *dot product* of vectors:
   - Define e so that
   $$e(g^{\vec{v}}, h^{\vec{w}}) = \hat{e}(g,h)^{\vec{v}\cdot\vec{w}}.$$
   - Define subgroups using orthogonal vectors:
   $$G_1 = \langle g^{\vec{v}} \rangle, G_2 = \langle g^{\vec{w}} \rangle, \quad H_1 = \langle h^{\vec{v}'} \rangle, H_2 = \langle h^{\vec{w}'} \rangle$$
   with $\vec{v} \cdot \vec{w}' = \vec{w} \cdot \vec{v}' = 0$.

Projecting and cancelling pairings on product groups

View group elements as vectors $g^{\vec{v}} = (g^{v_1}, g^{v_2})$.
Do linear algebra in the exponent.

1. *Projecting* pairing takes *tensor product* of vectors:
   - Define $e\colon G \times H \to G_t := \mathbb{G}_t^4$ to be vector of all 4 componentwise pairings $\hat{e}$ on $\mathbb{G}_1 \times \mathbb{G}_2$.
   - π_1, π_2, π_t do linear projection in the exponent (details in paper).
2. *Cancelling* pairing takes *dot product* of vectors:
   - Define e so that
     $$e(g^{\vec{v}}, h^{\vec{w}}) = \hat{e}(g, h)^{\vec{v}\cdot\vec{w}}.$$
   - Define subgroups using orthogonal vectors:
     $$G_1 = \langle g^{\vec{v}} \rangle, G_2 = \langle g^{\vec{w}} \rangle, \quad H_1 = \langle h^{\vec{v}'} \rangle, H_2 = \langle h^{\vec{w}'} \rangle$$
     with $\vec{v} \cdot \vec{w}' = \vec{w} \cdot \vec{v}' = 0$.

Projecting and cancelling pairings on product groups

View group elements as vectors $g^{\vec{v}} = (g^{v_1}, g^{v_2})$.
Do linear algebra in the exponent.

1. *Projecting* pairing takes *tensor product* of vectors:
   - Define $e\colon G \times H \to G_t := \mathbb{G}_t^4$ to be vector of all 4 componentwise pairings $\hat{e}$ on $\mathbb{G}_1 \times \mathbb{G}_2$.
   - π_1, π_2, π_t do linear projection in the exponent (details in paper).
2. *Cancelling* pairing takes *dot product* of vectors:
   - Define e so that
   $$e(g^{\vec{v}}, h^{\vec{w}}) = \hat{e}(g, h)^{\vec{v}\cdot\vec{w}}.$$
   - Define subgroups using orthogonal vectors:
   $$G_1 = \langle g^{\vec{v}} \rangle, G_2 = \langle g^{\vec{w}} \rangle, \quad H_1 = \langle h^{\vec{v}'} \rangle, H_2 = \langle h^{\vec{w}'} \rangle$$
   with $\vec{v} \cdot \vec{w}' = \vec{w} \cdot \vec{v}' = 0$.

How to convert a composite-order cryptosystem to prime-order groups

1. Write the system using our abstract group framework, with appropriate type of pairing.
   - Transfer to asymmetric groups for greatest generality.
2. Translate security assumption to general framework.
   - Check the security proof!
3. Instantiate system and assumption using groups G, H constructed from $\mathbb{G}_1, \mathbb{G}_2$.
   - e.g. generalized subgroup decision assumption instantiated as DDH.

Instantiating BGN Encryption in DDH groups $\mathbb{G}_1, \mathbb{G}_2$:

- PK: $G = \mathbb{G}_1^2$, $G_1 = \langle (g, g^x) \rangle$, $\hat{g} = (g^y, g^z)$, + similar in $H = \mathbb{G}_2^2$.
  SK: x, y, z + analogues for H.
- Encryption in G: encode msg using $\hat{g}$, blind with random elt of G_1.

$$Enc(m)\colon r \xleftarrow{\mathrm{R}} \mathbb{F}_p; \quad C = (g^y, g^z)^m (g, g^x)^r = (g^{ym+r}, g^{zm+xr})$$

  Encryption in H similar.
- Add by multiplying ciphertexts; multiply once by pairing ciphertexts.
  - Use projecting pairing e (vector of 4 pairings).
- Decryption in G:
  1. Compute $\pi_1(C) = (g^{ym+r})^x \cdot (g^{zm+xr})^{-1} = (g^{xy-z})^m$.
  2. Take discrete log base $\pi_1(\hat{g}) = g^{xy-z}$ (requires small message space).

  Decryption in H similar; decryption in $G_t = \mathbb{G}_t^4$ more complicated.

DDH in $\mathbb{G}_1, \mathbb{G}_2 \Rightarrow$ subgp decision in $(G, G_1, H, H_1, e) \Rightarrow$ semantic security.

Instantiating BGN Encryption in DDH groups $\mathbb{G}_1, \mathbb{G}_2$:

- PK: $G = \mathbb{G}_1^2$, $G_1 = \langle (g, g^x) \rangle$, $\hat{g} = (g^y, g^z)$, + similar in $H = \mathbb{G}_2^2$.
  SK: x, y, z + analogues for H.
- Encryption in G: encode msg using $\hat{g}$, blind with random elt of G_1.

$$Enc(m)\colon r \xleftarrow{\mathrm{R}} \mathbb{F}_p; \quad C = (g^y, g^z)^m (g, g^x)^r = (g^{ym+r}, g^{zm+xr})$$

  Encryption in H similar.
- Add by multiplying ciphertexts; multiply once by pairing ciphertexts.
  - Use projecting pairing e (vector of 4 pairings).
- Decryption in G:
  1. Compute $\pi_1(C) = (g^{ym+r})^x \cdot (g^{zm+xr})^{-1} = (g^{xy-z})^m$.
  2. Take discrete log base $\pi_1(\hat{g}) = g^{xy-z}$ (requires small message space).

  Decryption in H similar; decryption in $G_t = \mathbb{G}_t^4$ more complicated.

DDH in $\mathbb{G}_1, \mathbb{G}_2 \Rightarrow$ subgp decision in $(G, G_1, H, H_1, e) \Rightarrow$ semantic security.

Instantiating BGN Encryption in DDH groups $\mathbb{G}_1, \mathbb{G}_2$:

- PK: $G = \mathbb{G}_1^2$, $G_1 = \langle (g, g^x) \rangle$, $\hat{g} = (g^y, g^z)$, + similar in $H = \mathbb{G}_2^2$.
  SK: x, y, z + analogues for H.
- Encryption in G: encode msg using $\hat{g}$, blind with random elt of G_1.

$$Enc(m) \colon r \xleftarrow{\mathrm{R}} \mathbb{F}_p; \quad C = (g^y, g^z)^m (g, g^x)^r = (g^{ym+r}, g^{zm+xr})$$

  Encryption in H similar.
- Add by multiplying ciphertexts; multiply once by pairing ciphertexts.
  - Use projecting pairing e (vector of 4 pairings).
- Decryption in G:
  1. Compute $\pi_1(C) = (g^{ym+r})^x \cdot (g^{zm+xr})^{-1} = (g^{xy-z})^m$.
  2. Take discrete log base $\pi_1(\hat{g}) = g^{xy-z}$ (requires small message space).

  Decryption in H similar; decryption in $G_t = \mathbb{G}_t^4$ more complicated.

DDH in $\mathbb{G}_1, \mathbb{G}_2 \Rightarrow$ subgp decision in $(G, G_1, H, H_1, e) \Rightarrow$ semantic security.

Instantiating BGN Encryption in DDH groups $\mathbb{G}_1, \mathbb{G}_2$:

- PK: $G = \mathbb{G}_1^2$, $G_1 = \langle (g, g^x) \rangle$, $\hat{g} = (g^y, g^z)$, + similar in $H = \mathbb{G}_2^2$.
  SK: x, y, z + analogues for H.
- Encryption in G: encode msg using $\hat{g}$, blind with random elt of G_1.

$$Enc(m)\colon r \xleftarrow{\mathrm{R}} \mathbb{F}_p; \quad C = (g^y, g^z)^m (g, g^x)^r = (g^{ym+r}, g^{zm+xr})$$

  Encryption in H similar.
- Add by multiplying ciphertexts; multiply once by pairing ciphertexts.
  - Use projecting pairing e (vector of 4 pairings).
- Decryption in G:
  1. Compute $\pi_1(C) = (g^{ym+r})^x \cdot (g^{zm+xr})^{-1} = (g^{xy-z})^m$.
  2. Take discrete log base $\pi_1(\hat{g}) = g^{xy-z}$ (requires small message space).

  Decryption in H similar; decryption in $G_t = \mathbb{G}_t^4$ more complicated.

DDH in $\mathbb{G}_1, \mathbb{G}_2 \Rightarrow$ subgp decision in $(G, G_1, H, H_1, e) \Rightarrow$ semantic security.

Instantiating BGN Encryption in DDH groups $\mathbb{G}_1, \mathbb{G}_2$:

- PK: $G = \mathbb{G}_1^2$, $G_1 = \langle (g, g^x) \rangle$, $\hat{g} = (g^y, g^z)$, + similar in $H = \mathbb{G}_2^2$.
  SK: x, y, z + analogues for H.
- Encryption in G: encode msg using $\hat{g}$, blind with random elt of G_1.

$$Enc(m)\colon r \xleftarrow{\mathrm{R}} \mathbb{F}_p; \quad C = (g^y, g^z)^m (g, g^x)^r = (g^{ym+r}, g^{zm+xr})$$

  Encryption in H similar.
- Add by multiplying ciphertexts; multiply once by pairing ciphertexts.
  - Use projecting pairing e (vector of 4 pairings).
- Decryption in G:
  1. Compute $\pi_1(C) = (g^{ym+r})^x \cdot (g^{zm+xr})^{-1} = (g^{xy-z})^m$.
  2. Take discrete log base $\pi_1(\hat{g}) = g^{xy-z}$ (requires small message space).

  Decryption in H similar; decryption in $G_t = \mathbb{G}_t^4$ more complicated.

DDH in $\mathbb{G}_1, \mathbb{G}_2 \Rightarrow$ subgp decision in $(G, G_1, H, H_1, e) \Rightarrow$ semantic security.

Other systems

We also applied our conversion process to BSW traitor tracing and KSW attribute-based encryption.

- Groups become smaller and pairing computations become much faster.
- Security assumptions remain of comparable complexity.
- Efficiency improvement is greater at higher security levels:

	Bit size of BGN ciphertexts	
Security level	composite-order	prime-order
80-bit	1024	1020
128-bit	3072	1536
256-bit	15360	6400

Conclusion: Most things that can be done using composite-order bilinear groups can be done more efficiently using prime-order bilinear groups.

Other systems

We also applied our conversion process to BSW traitor tracing and KSW attribute-based encryption.

- Groups become smaller and pairing computations become much faster.
- Security assumptions remain of comparable complexity.
- Efficiency improvement is greater at higher security levels:

	Bit size of BGN ciphertexts	
Security level	composite-order	prime-order
80-bit	1024	1020
128-bit	3072	1536
256-bit	15360	6400

Conclusion: Most things that can be done using composite-order bilinear groups can be done more efficiently using prime-order bilinear groups.

Other systems

We also applied our conversion process to BSW traitor tracing and KSW attribute-based encryption.

- Groups become smaller and pairing computations become much faster.
- Security assumptions remain of comparable complexity.
- Efficiency improvement is greater at higher security levels:

	Bit size of BGN ciphertexts	
Security level	composite-order	prime-order
80-bit	1024	1020
128-bit	3072	1536
256-bit	15360	6400

Conclusion: Most things that can be done using composite-order bilinear groups can be done more efficiently using prime-order bilinear groups.