

# On the Static Diffie-Hellman Problem on Elliptic Curves over Extension Fields

Robert Granger

rgranger@computing.dcu.ie  
Claude Shannon Institute, UCD and DCU, Ireland

Workshop on Elliptic Curve Computation, October 18-22,  
2010

# Outline

- 1 Motivation - Exotic Security Assumptions in Cryptography
  - The Static Diffie-Hellman Problem
  - Other Assumptions
  - A Contrived Example
- 2 Main Algorithm and Results
  - Algorithm Overview
  - Potentially Vulnerable Curves
  - Simulation results
- 3 Oracle-assisted Static DHP for binary curves

# Provable Security 101

# Provable Security 101

- Cryptographic protocols require security assurances established via a reductionist security argument — also known as a proof of security

# Provable Security 101

- Cryptographic protocols require security assurances established via a reductionist security argument — also known as a proof of security
- These take the form ‘If one can break the protocol in some way, then one can solve a related computational problem’, which is assumed to be hard

# Provable Security 101

- Cryptographic protocols require security assurances established via a reductionist security argument — also known as a proof of security
- These take the form ‘If one can break the protocol in some way, then one can solve a related computational problem’, which is assumed to be hard

In an ideal world:

# Provable Security 101

- Cryptographic protocols require security assurances established via a reductionist security argument — also known as a proof of security
- These take the form ‘If one can break the protocol in some way, then one can solve a related computational problem’, which is assumed to be hard

In an ideal world:

- Such assumptions should be as hard to solve as the underlying primitive, such as the RSA function or the DLP

# Provable Security 101

- Cryptographic protocols require security assurances established via a reductionist security argument — also known as a proof of security
- These take the form ‘If one can break the protocol in some way, then one can solve a related computational problem’, which is assumed to be hard

In an ideal world:

- Such assumptions should be as hard to solve as the underlying primitive, such as the RSA function or the DLP
- There should be an equivalence result, so that if one breaks the hard problem, one can also break the protocol



# Is Cryptography always ideal?

In 'Another look at non-standard discrete log and Diffie-Hellman problems' (2007), Koblitz and Menezes studied a set of problems in the Jacobian of small genus hyperelliptic curves

# Is Cryptography always ideal?

In 'Another look at non-standard discrete log and Diffie-Hellman problems' (2007), Koblitz and Menezes studied a set of problems in the Jacobian of small genus hyperelliptic curves

- *Delayed Target* DLP/DHP, *One-More* DLP/DHP, and DLP1/DHP1 - DLP/DHP being well-studied

# Is Cryptography always ideal?

In 'Another look at non-standard discrete log and Diffie-Hellman problems' (2007), Koblitz and Menezes studied a set of problems in the Jacobian of small genus hyperelliptic curves

- *Delayed Target* DLP/DHP, *One-More* DLP/DHP, and DLP1/DHP1 - DLP/DHP being well-studied
- Some are equivalent to breaking protocols, some are not

# Is Cryptography always ideal?

In 'Another look at non-standard discrete log and Diffie-Hellman problems' (2007), Koblitz and Menezes studied a set of problems in the Jacobian of small genus hyperelliptic curves

- *Delayed Target* DLP/DHP, *One-More* DLP/DHP, and DLP1/DHP1 - DLP/DHP being well-studied
- Some are equivalent to breaking protocols, some are not
- Using 'Index Calculus' or generic arguments showed that some are easier than DLP - hardness separation

# Is Cryptography always ideal?

In 'Another look at non-standard discrete log and Diffie-Hellman problems' (2007), Koblitz and Menezes studied a set of problems in the Jacobian of small genus hyperelliptic curves

- *Delayed Target* DLP/DHP, *One-More* DLP/DHP, and DLP1/DHP1 - DLP/DHP being well-studied
- Some are equivalent to breaking protocols, some are not
- Using 'Index Calculus' or generic arguments showed that some are easier than DLP - hardness separation
- Argued that problems which are either interactive or have complicated inputs yield weaknesses

# Is Cryptography always ideal?

In 'Another look at non-standard discrete log and Diffie-Hellman problems' (2007), Koblitz and Menezes studied a set of problems in the Jacobian of small genus hyperelliptic curves

- *Delayed Target* DLP/DHP, *One-More* DLP/DHP, and DLP1/DHP1 - DLP/DHP being well-studied
- Some are equivalent to breaking protocols, some are not
- Using 'Index Calculus' or generic arguments showed that some are easier than DLP - hardness separation
- Argued that problems which are either interactive or have complicated inputs yield weaknesses
- Security assurances provided by such protocols should be reassessed/are difficult to assess

# Diffie-Hellman Key Agreement

Let  $\mathbb{G}$  be a cyclic group of prime order  $r$  with generator  $g$ .

# Diffie-Hellman Key Agreement

Let  $\mathbb{G}$  be a cyclic group of prime order  $r$  with generator  $g$ .

- Alice chooses  $x \xleftarrow{R} \mathbb{Z}_r$ , computes  $g^x$  and sends to Bob



# Diffie-Hellman Key Agreement

Let  $\mathbb{G}$  be a cyclic group of prime order  $r$  with generator  $g$ .

- Alice chooses  $x \xleftarrow{R} \mathbb{Z}_r$ , computes  $g^x$  and sends to Bob
- Bob chooses  $y \xleftarrow{R} \mathbb{Z}_r$ , computes  $g^y$  and sends to Alice

# Diffie-Hellman Key Agreement

Let  $\mathbb{G}$  be a cyclic group of prime order  $r$  with generator  $g$ .

- Alice chooses  $x \xleftarrow{R} \mathbb{Z}_r$ , computes  $g^x$  and sends to Bob
- Bob chooses  $y \xleftarrow{R} \mathbb{Z}_r$ , computes  $g^y$  and sends to Alice
- Alice computes  $(g^y)^x$ , Bob computes  $(g^x)^y$  to give shared secret  $g^{xy}$

# Diffie-Hellman Key Agreement

Let  $\mathbb{G}$  be a cyclic group of prime order  $r$  with generator  $g$ .

- Alice chooses  $x \xleftarrow{R} \mathbb{Z}_r$ , computes  $g^x$  and sends to Bob
- Bob chooses  $y \xleftarrow{R} \mathbb{Z}_r$ , computes  $g^y$  and sends to Alice
- Alice computes  $(g^y)^x$ , Bob computes  $(g^x)^y$  to give shared secret  $g^{xy}$

A fundamental security requirement of DH Key Agreement is that the *Computational Diffie-Hellman* problem should be hard:

## Definition

(CDH): Given  $g$  and random  $g^x$  and  $g^y$ , find  $g^{xy}$

# The Static Diffie-Hellman Problem (Static DHP)

Suppose in DH Key Agreement Alice repeatedly reuses  $x = d$ .

# The Static Diffie-Hellman Problem (Static DHP)

Suppose in DH Key Agreement Alice repeatedly reuses  $x = d$ .

## Definition

(Static DHP <sub>$d$</sub> ): Given fixed  $g$  and  $g^d$ , and random  $g^y$ , find  $g^{dy}$

# The Static Diffie-Hellman Problem (Static DHP)

Suppose in DH Key Agreement Alice repeatedly reuses  $x = d$ .

## Definition

(Static DHP <sub>$d$</sub> ): Given fixed  $g$  and  $g^d$ , and random  $g^y$ , find  $g^{dy}$

- Set of problem instances in Static DHP is a tiny subset of CDH problem instances

# The Static Diffie-Hellman Problem (Static DHP)

Suppose in DH Key Agreement Alice repeatedly reuses  $x = d$ .

## Definition

(Static DHP <sub>$d$</sub> ): Given fixed  $g$  and  $g^d$ , and random  $g^y$ , find  $g^{dy}$

- Set of problem instances in Static DHP is a tiny subset of CDH problem instances
- Not *a priori* clear that these instances should be hard, even if CDH is hard

# The Static Diffie-Hellman Problem (Static DHP)

Suppose in DH Key Agreement Alice repeatedly reuses  $x = d$ .

## Definition

(Static DHP <sub>$d$</sub> ): Given fixed  $g$  and  $g^d$ , and random  $g^y$ , find  $g^{dy}$

- Set of problem instances in Static DHP is a tiny subset of CDH problem instances
- Not *a priori* clear that these instances should be hard, even if CDH is hard
- Can arise as an efficiency measure during multiple DH key agreements



# The Static Diffie-Hellman Problem (Static DHP)

Suppose in DH Key Agreement Alice repeatedly reuses  $x = d$ .

## Definition

(Static DHP <sub>$d$</sub> ): Given fixed  $g$  and  $g^d$ , and random  $g^y$ , find  $g^{dy}$

- Set of problem instances in Static DHP is a tiny subset of CDH problem instances
- Not *a priori* clear that these instances should be hard, even if CDH is hard
- Can arise as an efficiency measure during multiple DH key agreements
- Also arises in textbook El Gamal encryption, Ford-Kaliski key retrieval, and Chaum-Van Antwerpen's undeniable signatures

## Static DHP<sub>d</sub> example: textbook El Gamal

- Alice has public key  $g^d$ . To encrypt a message  $m$ , Bob picks a random  $x \xleftarrow{R} \mathbb{Z}_r$  and computes

$$c = (c_1, c_2) = (g^x, mg^{dx})$$

## Static DHP<sub>d</sub> example: textbook El Gamal

- Alice has public key  $g^d$ . To encrypt a message  $m$ , Bob picks a random  $x \xleftarrow{R} \mathbb{Z}_r$  and computes

$$c = (c_1, c_2) = (g^x, mg^{dx})$$

- To decrypt Alice computes  $m = c_2/c_1^d$ . So if one can compute  $g^{dx}$  for any  $g^x$  one can decrypt

## Static DHP<sub>d</sub> example: textbook El Gamal

- Alice has public key  $g^d$ . To encrypt a message  $m$ , Bob picks a random  $x \xleftarrow{R} \mathbb{Z}_r$  and computes

$$c = (c_1, c_2) = (g^x, mg^{dx})$$

- To decrypt Alice computes  $m = c_2/c_1^d$ . So if one can compute  $g^{dx}$  for any  $g^x$  one can decrypt
- Furthermore, in a chosen-ciphertext attack an adversary has access to a decryption oracle

## Static DHP<sub>d</sub> example: textbook El Gamal

- Alice has public key  $g^d$ . To encrypt a message  $m$ , Bob picks a random  $x \xleftarrow{R} \mathbb{Z}_r$  and computes

$$c = (c_1, c_2) = (g^x, mg^{dx})$$

- To decrypt Alice computes  $m = c_2/c_1^d$ . So if one can compute  $g^{dx}$  for any  $g^x$  one can decrypt
- Furthermore, in a chosen-ciphertext attack an adversary has access to a decryption oracle
- If adversary chooses  $c = (g^x, c_2)$  the decryption oracle returns  $m = c_2/g^{dx}$

## Static DHP<sub>d</sub> example: textbook El Gamal

- Alice has public key  $g^d$ . To encrypt a message  $m$ , Bob picks a random  $x \xleftarrow{R} \mathbb{Z}_r$  and computes

$$c = (c_1, c_2) = (g^x, mg^{dx})$$

- To decrypt Alice computes  $m = c_2/c_1^d$ . So if one can compute  $g^{dx}$  for any  $g^x$  one can decrypt
- Furthermore, in a chosen-ciphertext attack an adversary has access to a decryption oracle
- If adversary chooses  $c = (g^x, c_2)$  the decryption oracle returns  $m = c_2/g^{dx}$
- Adversary computes  $g^{dx} = c_2/m$ , which solves the Static DHP<sub>d</sub> for instance  $g^x$ , hence the adversary has access to a **Static DHP<sub>d</sub> oracle**

# Static DHP<sub>d</sub> oracle

## Definition

(Static DHP<sub>d</sub> oracle). Let  $\mathbb{G}$  be a cyclic group of prime order  $r$ , written additively. For a fixed base element  $P \in \mathbb{G}$  and a fixed element  $Q \in \mathbb{G}$  let  $d \in \mathbb{Z}_r$  be such that  $Q = dP$ . Then a Static DHP<sub>d</sub> oracle (wrt  $\mathbb{G}$ ) computes the function  $\delta : \mathbb{G} \rightarrow \mathbb{G}$  where:

$$\delta(X) = dX.$$

# Static DHP<sub>d</sub> oracle

## Definition

(Static DHP<sub>d</sub> oracle). Let  $\mathbb{G}$  be a cyclic group of prime order  $r$ , written additively. For a fixed base element  $P \in \mathbb{G}$  and a fixed element  $Q \in \mathbb{G}$  let  $d \in \mathbb{Z}_r$  be such that  $Q = dP$ . Then a Static DHP<sub>d</sub> oracle (wrt  $\mathbb{G}$ ) computes the function  $\delta : \mathbb{G} \rightarrow \mathbb{G}$  where:

$$\delta(X) = dX.$$

Likewise, a Static DHP<sub>d</sub> algorithm is said to be *oracle-assisted* if during an initial learning phase, it can make a number of Static DHP<sub>d</sub> queries, after which, given a previously unseen challenge element  $X$ , it outputs  $dX$ .



# The Static DHP - inception and 1st result

Introduced by Brown and Gallant in 2004, who gave a reduction from the DLP for  $d$  to the Static DHP $_d$

# The Static DHP - inception and 1st result

Introduced by Brown and Gallant in 2004, who gave a reduction from the DLP for  $d$  to the Static DHP $_d$

- Hence if the DLP for  $d$  is hard, then so is the Static DHP $_d$

# The Static DHP - inception and 1st result

Introduced by Brown and Gallant in 2004, who gave a reduction from the DLP for  $d$  to the Static DHP $_d$

- Hence if the DLP for  $d$  is hard, then so is the Static DHP $_d$
- Equivalently, given access to a Static DHP $_d$  oracle, one can find the associated DLP

# The Static DHP - inception and 1st result

Introduced by Brown and Gallant in 2004, who gave a reduction from the DLP for  $d$  to the Static DHP $_d$

- Hence if the DLP for  $d$  is hard, then so is the Static DHP $_d$
- Equivalently, given access to a Static DHP $_d$  oracle, one can find the associated DLP

## Theorem

*Let  $r = uv + 1$ . Then  $d$  can be found with  $u$  calls to a Static DHP $_d$  oracle, and off-line computational work of about  $(\sqrt{u} + \sqrt{v})$  group operations.*

# DLP to Static DHP<sub>d</sub> reduction

---

## ALGORITHM 1: DLP to Static DHP<sub>d</sub> reduction

---

INPUT:  $P, dP \in \mathbb{G}$  with  $\mathbb{G}$  cyclic of prime order  
and  $|\mathbb{G}| = uv + 1$

OUTPUT:  $d$

1. Find a generator  $g \in \mathbb{F}_p^\times$
2. Compute  $w = g^u$  and  $d^u P$  via  $u$  calls to oracle
3. Let  $m_v = \lceil \sqrt{v} \rceil$
4. Find  $0 \leq u_1, v_1 < m_v$  such that  $w^{-u_1} d^u P = w^{m_v v_1} P$
5. Let  $k_0 = m_v v_1 + u_1$  so that  $d^u = w^{k_0}$
6. Let  $m_u = \lceil \sqrt{u} \rceil$
7. Find  $0 \leq u_2, v_2 < m_u$  such that  $g^{-u_2 v} dP = g^{k_0 + m_u v_2 v} P$
8. Return  $g^{k_0 + (m_u v_2 + u_2)v}$

# DLP to Static DHP<sub>d</sub> reduction

- The complexity of the attack is minimised when  $u \approx r^{1/3}$

# DLP to Static DHP<sub>d</sub> reduction

- The complexity of the attack is minimised when  $u \approx r^{1/3}$
- For the three schemes mentioned, Brown and Gallant showed that a system entity acts as a Static DHP<sub>d</sub> oracle, transforming their reduction into a DLP solver

## DLP to Static DHP<sub>d</sub> reduction

- The complexity of the attack is minimised when  $u \approx r^{1/3}$
- For the three schemes mentioned, Brown and Gallant showed that a system entity acts as a Static DHP<sub>d</sub> oracle, transforming their reduction into a DLP solver
- Depending on the factorisation of  $r - 1$ , can lead to a real attack which is quicker than solving the DLP



## DLP to Static DHP<sub>d</sub> reduction

- The complexity of the attack is minimised when  $u \approx r^{1/3}$
- For the three schemes mentioned, Brown and Gallant showed that a system entity acts as a Static DHP<sub>d</sub> oracle, transforming their reduction into a DLP solver
- Depending on the factorisation of  $r - 1$ , can lead to a real attack which is quicker than solving the DLP

Attack was rediscovered by Cheon in 2006, when the requisite information is provided in the guise of the  $l$ -Strong DHP:

## DLP to Static DHP<sub>d</sub> reduction

- The complexity of the attack is minimised when  $u \approx r^{1/3}$
- For the three schemes mentioned, Brown and Gallant showed that a system entity acts as a Static DHP<sub>d</sub> oracle, transforming their reduction into a DLP solver
- Depending on the factorisation of  $r - 1$ , can lead to a real attack which is quicker than solving the DLP

Attack was rediscovered by Cheon in 2006, when the requisite information is provided in the guise of the  $l$ -Strong DHP:

### Definition

$l$ -Strong Diffie-Hellman problem: Given  $P$  and  $d^i P$  in  $\mathbb{G}$  for  $i = 1, 2, \dots, l$ , compute  $d^{l+1} P$ .

## DLP to $l$ -Strong DHP reduction

- Cheon also formulated an algorithm for the  $l$ -Strong DHP when  $l \mid (r + 1)$

## DLP to $l$ -Strong DHP reduction

- Cheon also formulated an algorithm for the  $l$ -Strong DHP when  $l \mid (r + 1)$
- Brown-Gallant reduction and Cheon's algorithm can be seen as using the DLP to DHP reduction due to den Boer, Maurer, Wolf et al, but with limited access to CDH oracle

## DLP to $l$ -Strong DHP reduction

- Cheon also formulated an algorithm for the  $l$ -Strong DHP when  $l \mid (r + 1)$
- Brown-Gallant reduction and Cheon's algorithm can be seen as using the DLP to DHP reduction due to den Boer, Maurer, Wolf et al, but with limited access to CDH oracle
- For the  $l$ -Strong DHP, security proofs were in one direction only, so Cheon's algorithm does not break any protocols

## DLP to $l$ -Strong DHP reduction

- Cheon also formulated an algorithm for the  $l$ -Strong DHP when  $l \mid (r + 1)$
- Brown-Gallant reduction and Cheon's algorithm can be seen as using the DLP to DHP reduction due to den Boer, Maurer, Wolf et al, but with limited access to CDH oracle
- For the  $l$ -Strong DHP, security proofs were in one direction only, so Cheon's algorithm does not break any protocols
- For Boneh-Boyen signatures, in 2009 Jao and Yoshida gave a reduction in the reverse direction, thus strengthening the security proof, and giving an attack with complexity  $O(r^{2/5+\epsilon})$  if  $O(r^{1/5+\epsilon})$  signatures queries may be performed

# Delayed Target DHP

## Definition

A solver is given initial access to a Static DHP<sub>*d*</sub> oracle for the element  $Q = dP \in \mathbb{G}$ ; when the oracle is removed, the solver is given a random challenge  $X \in \mathbb{G}$  and must solve the CDH for input  $(Q, X)$ , i.e., output  $dX$ .

# Delayed Target DHP

## Definition

A solver is given initial access to a Static DHP<sub>*d*</sub> oracle for the element  $Q = dP \in \mathbb{G}$ ; when the oracle is removed, the solver is given a random challenge  $X \in \mathbb{G}$  and must solve the CDH for input  $(Q, X)$ , i.e., output  $dX$ .

- Situation identical to oracle-assisted Static DHP



# Delayed Target DHP

## Definition

A solver is given initial access to a Static DHP<sub>d</sub> oracle for the element  $Q = dP \in \mathbb{G}$ ; when the oracle is removed, the solver is given a random challenge  $X \in \mathbb{G}$  and must solve the CDH for input  $(Q, X)$ , i.e., output  $dX$ .

- Situation identical to oracle-assisted Static DHP
- Described by Freeman, 2005 — ‘Pairing-based identification schemes’

# Delayed Target DHP

## Definition

A solver is given initial access to a Static DHP<sub>*d*</sub> oracle for the element  $Q = dP \in \mathbb{G}$ ; when the oracle is removed, the solver is given a random challenge  $X \in \mathbb{G}$  and must solve the CDH for input  $(Q, X)$ , i.e., output  $dX$ .

- Situation identical to oracle-assisted Static DHP
- Described by Freeman, 2005 — ‘Pairing-based identification schemes’
- Security of scheme equivalent to Delayed Target DHP

# Delayed Target DHP

## Definition

A solver is given initial access to a Static DHP<sub>d</sub> oracle for the element  $Q = dP \in \mathbb{G}$ ; when the oracle is removed, the solver is given a random challenge  $X \in \mathbb{G}$  and must solve the CDH for input  $(Q, X)$ , i.e., output  $dX$ .

- Situation identical to oracle-assisted Static DHP
- Described by Freeman, 2005 — ‘Pairing-based identification schemes’
- Security of scheme equivalent to Delayed Target DHP
- Koblitz-Menezes also studied the *Delayed Target DLP*

# The oracle-assisted Static DHP/Delayed Target DHP

Koblitz-Menezes used index calculus methodology:

# The oracle-assisted Static DHP/Delayed Target DHP

Koblitz-Menezes used index calculus methodology:

- Construct a factor base  $\mathcal{F}$  over which a non-negligible proportion of group elements factor

# The oracle-assisted Static DHP/Delayed Target DHP

Koblitz-Menezes used index calculus methodology:

- Construct a factor base  $\mathcal{F}$  over which a non-negligible proportion of group elements factor
- Call the Static DHP<sub>d</sub> oracle  $\delta$  on all  $f \in \mathcal{F}$

# The oracle-assisted Static DHP/Delayed Target DHP

Koblitz-Menezes used index calculus methodology:

- Construct a factor base  $\mathcal{F}$  over which a non-negligible proportion of group elements factor
- Call the Static DHP<sub>d</sub> oracle  $\delta$  on all  $f \in \mathcal{F}$
- For a target element  $X$  attempt to write random multiples  $aX$  as a sum of elements of  $\mathcal{F}$ , i.e.,  $aX = P_1 + \dots + P_n$

# The oracle-assisted Static DHP/Delayed Target DHP

Koblitz-Menezes used index calculus methodology:

- Construct a factor base  $\mathcal{F}$  over which a non-negligible proportion of group elements factor
- Call the Static DHP $_d$  oracle  $\delta$  on all  $f \in \mathcal{F}$
- For a target element  $X$  attempt to write random multiples  $aX$  as a sum of elements of  $\mathcal{F}$ , i.e.,  $aX = P_1 + \dots + P_n$
- Then  $dX = (a^{-1} \bmod r)(\delta(P_1) + \dots + \delta(P_n))$



# The oracle-assisted Static DHP/Delayed Target DHP

Koblitz-Menezes used index calculus methodology:

- Construct a factor base  $\mathcal{F}$  over which a non-negligible proportion of group elements factor
- Call the Static DHP $_d$  oracle  $\delta$  on all  $f \in \mathcal{F}$
- For a target element  $X$  attempt to write random multiples  $aX$  as a sum of elements of  $\mathcal{F}$ , i.e.,  $aX = P_1 + \dots + P_n$
- Then  $dX = (a^{-1} \bmod r)(\delta(P_1) + \dots + \delta(P_n))$

Used for finite fields and small genus hyperelliptic curves —  
hardness separation from DLP

# The Static DHP - JLNT

Joux, Naccache and Thomé showed that initial subexponential access to an  $e$ -th root oracle in RSA enables later  $e$ -th root computations — faster than one can factor the modulus

# The Static DHP - JLNT

Joux, Naccache and Thomé showed that initial subexponential access to an  $e$ -th root oracle in RSA enables later  $e$ -th root computations — faster than one can factor the modulus

- Ports easily over to Static DHP <sub>$d$</sub>  in finite fields (+Lercier)

# The Static DHP - JLNT

Joux, Naccache and Thomé showed that initial subexponential access to an  $e$ -th root oracle in RSA enables later  $e$ -th root computations — faster than one can factor the modulus

- Ports easily over to Static DHP $_d$  in finite fields (+Lercier)
- The  $L_{q^n}(1/3, \sqrt[3]{x})$  complexities of the JLNT algorithm are

variant	oracle access	learning phase	post-learning phase
FFS	4/9	-	4/9
NFS-HD	48/91	384/91	384/91
NFS	4/9	32/9	3

# The Static DHP - JLNT

Joux, Naccache and Thomé showed that initial subexponential access to an  $e$ -th root oracle in RSA enables later  $e$ -th root computations — faster than one can factor the modulus

- Ports easily over to Static DHP $_d$  in finite fields (+Lercier)
- The  $L_{q^n}(1/3, \sqrt[3]{x})$  complexities of the JLNT algorithm are

variant	oracle access	learning phase	post-learning phase
FFS	4/9	-	4/9
NFS-HD	48/91	384/91	384/91
NFS	4/9	32/9	3

- Observe that each is faster than the DLP in the corresponding fields

## Index calculus example: *Delayed Target* DHP

Let  $H(\mathbb{F}_q)$  be a genus  $g$  hyperelliptic curve and  $Jac_H(\mathbb{F}_q)$  its Jacobian.

- Let  $\mathcal{F}$  be a proportion  $q^\alpha$  of degree one divisors for  $0 < \alpha \leq 1$ .
- Call the Static DHP $_d$  oracle for  $Q = dP$  for all  $D \in \mathcal{F}$ .
- Prob. random  $aX$  factors over  $\mathcal{F}$  is  $q^{g(\alpha-1)}/g!$
- Hence expected number of trials to obtain an  $\mathcal{F}$ -smooth element  $aX$  is  $q^{g(1-\alpha)}g!$
- Balancing this with the oracle calls gives

$$\alpha = (g + \log_q g!)/(g + 1) \approx 1 - 1/(g + 1)$$

# Index calculus example: *Delayed Target* DHP

For DLP, there are four basic variants:

- Gaudry (2000): basic index calculus —  $O(q^2)$
- Harley (2000): reduce factor base —  $O(q^{2-2/(g+1)})$
- Thériault (2003): large-prime variation —  $O(q^{2-2/(g+1/2)})$
- GTTD (2007): double large-prime variation —  $O(q^{2-2/g})$

The *Delayed Target* DHP algorithm is  $O(q^{1-1/(g+1)})$  — the square root of Harley's algorithm:

- No linear algebra
- Only one relation so can only balance the two stages

## Index calculus example: *Delayed Target* DHP

For DLP, there are four basic variants:

- Gaudry (2000): basic index calculus —  $O(q^2)$
- Harley (2000): reduce factor base —  $O(q^{2-2/(g+1)})$
- Thériault (2003): large-prime variation —  $O(q^{2-2/(g+1/2)})$
- GTTD (2007): double large-prime variation —  $O(q^{2-2/g})$

The *Delayed Target* DHP algorithm is  $O(q^{1-1/(g+1)})$  — the square root of Harley's algorithm:

- No linear algebra
- Only one relation so can only balance the two stages

**Question: For  $g = 1$  have  $O(q^{1/2})$ , so can we do better?**



# A contrived example: oracle-assisted pairing inversion

Let  $\mathbb{G}_1, \mathbb{G}_2$  and  $\mathbb{G}_T$  be cyclic groups of prime order  $r$ , and let

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

be a non-degenerate bilinear pairing.

- Verheul considered the consequences of the existence of an efficiently computable, injective homomorphism  $\psi : \mathbb{G}_T \longrightarrow \mathbb{G}_1$  when  $\mathbb{G}_1 = \mathbb{G}_2$
- If  $\psi$  is efficiently computable, can efficiently solve CDH in  $\mathbb{G}_1, \mathbb{G}_2$  and  $\mathbb{G}_T$

# Oracle-assisted pairing inversion

Galbriath, Hess and Vercauteren (2008) considered the following two problems when  $\mathbb{G}_1 \neq \mathbb{G}_2$ :

## Definition

The Fixed Argument Pairing Inversion 1 (FAPI-1) problem:  
Given  $P_1 \in \mathbb{G}_1$  and  $z \in \mathbb{G}_T$ , find  $P_2 \in \mathbb{G}_2$  s.t.  $e(P_1, P_2) = z$ .

The Fixed Argument Pairing Inversion 2 (FAPI-2) problem:  
Given  $P_2 \in \mathbb{G}_2$  and  $z \in \mathbb{G}_T$ , find  $P_1 \in \mathbb{G}_1$  s.t.  $e(P_1, P_2) = z$ .

- If can solve FAPI-1 and 2, can solve CDH in  $\mathbb{G}_1, \mathbb{G}_2$  and  $\mathbb{G}_T$
- Given  $P, aP, bP \in \mathbb{G}_1$ , fix  $Q \in \mathbb{G}_2$  and let  $z = e(P, Q)$ .  
Compute  $z^b = e(bP, Q)$ ,  $\text{FAPI-1}(P, z^b) = bQ$ ,  
 $e(aP, bQ) = z^{ab}$ , then  $\text{FAPI-2}(Q, z^{ab}) = abP$ .

# Oracle-assisted pairing inversion

GHV also showed that if one has an efficiently computable homomorphism  $\psi : \mathbb{G}_2 \rightarrow \mathbb{G}_1$  and access to a FAPI-1 oracle, then the same conclusion holds.

Some natural questions arise:

- In this case, what if one only has initial access to a FAPI-1 oracle? Can one solve further FAPI-1 instances — *Delayed Target FAPI-1 problem*
- Does this problem feature in the security of any protocol?
- What other security implications are there?
- Can FAPI-1 and no  $\psi$  aid in solving Static DHP?

## Delayed Target FAPI-1 problem

- Since  $\mathbb{G}_T \subset \mathbb{F}_{p^k}$ , natural (only?) idea is to use factor base in  $\mathbb{F}_{p^k}$  as in JLNT
- First problem is that in general elements of factor base in  $\mathbb{F}_{p^k}$  are not in  $G_T$ .
- Second problem is that one can't invert pairing!

However, can do the following:

- Compute  $Q_i = \text{FAPI-1}(P, p_i^{(p^k-1)/r})$  for each  $p_i \in \mathcal{F}$
- Want to compute  $\text{FAPI-1}(P, z)$  for  $z \in \mathbb{G}_T$
- Write  $z = \prod p_i^{\alpha_i}$  where  $p_i \in \mathcal{F}$  using JLNT
- Assuming  $((p^k - 1)/r, r) = 1$ , one has

$$\text{FAPI-1}(P, z) = (((p^k - 1)/r)^{-1} \bmod r) \sum \alpha_i Q_i$$

## Delayed Target FAPI-1 problem

This solves the FAPI-1 for  $z$  since

$$\begin{aligned} & e(P, (((p^k - 1)/r)^{-1} \bmod r) \sum \alpha_j Q_j) \\ &= \prod e(P, Q_j)^{\alpha_j (((p^k - 1)/r)^{-1} \bmod r)} \\ &= \prod p_j^{(((p^k - 1)/r) ((p^k - 1)/r)^{-1} \bmod r) \alpha_j} = z \end{aligned}$$

- Hence can solve *any* further FAPI-1 problem with the complexity of JLNT Static DHP algorithm
- For curves with  $\psi : \mathbb{G}_2 \rightarrow \mathbb{G}_1$  can then solve any further CDH in  $\mathbb{G}_1, \mathbb{G}_2$  and  $\mathbb{G}_T$  faster than DLP
- Using DLP to DHP reduction, for suitable parameters can solve DLP faster

## *Delayed Target* FAPI-1 problem

- FAPI-i allows one to use factorisation in an auxilliary group to form a factor base in  $G_i$
- Only works if can map both ways
- Assumptions too strong to solve Static DHP but not CDH, so what if no  $\psi$ ?
- Can't solve Static DHP without efficiently computable  $\psi$ , as result is in the wrong group
- Academic anyhow as no known way to implement FAPI-i

## *Delayed Target* FAPI-1 problem

- FAPI- $i$  allows one to use factorisation in an auxiliary group to form a factor base in  $G_i$
- Only works if can map both ways
- Assumptions too strong to solve Static DHP but not CDH, so what if no  $\psi$ ?
- Can't solve Static DHP without efficiently computable  $\psi$ , as result is in the wrong group
- Academic anyhow as no known way to implement FAPI- $i$

Realised that for ECs over extension fields, already have native factorisation via Gaudry/Semaev idea, so can use the Menezes-Koblitz methodology directedly.

# Algorithm Overview

Let  $E : Y^2 = X^3 + aX + b$ , over a field  $\mathbb{F}_{q^n}$  with  $\text{char}(\mathbb{F}_q) > 3$ .

- Let  $\mathcal{F} = \{P = (x, y) \in E \text{ s.t. } x \in \mathbb{F}_q\}$
- For all  $P \in \mathcal{F}$  compute  $\delta(P) = dP$
- For a given  $R \in E(\mathbb{F}_{q^n})$  add random linear combinations  $P_r$  of elements of  $\mathcal{F}$  to  $R$  until it can be written

$$R + P_r = P_1 + \dots + P_n$$

- Then  $dR = \delta(P_1) + \dots + \delta(P_n) - \delta(P_r)$



# Semaev's summation polynomials

- For  $m \geq 2$  define  $f_m = f_m(X_1, \dots, X_m)$  by the following property. Let  $x_1, \dots, x_m \in \overline{\mathbb{F}}_q$ , then  $f_m(x_1, \dots, x_m) = 0$  is equivalent to

$$\exists y_1, \dots, y_m \in \overline{\mathbb{F}}_q \mid (x_i, y_i) \in E \quad \text{and} \\ (x_1, y_1) + \dots + (x_m, y_m) = \mathcal{O} \in E(\overline{\mathbb{F}}_q)$$

- $f_2(X_1, X_2) = X_1 - X_2$ , and  $f_3(X_1, X_2, X_3) =$

$$(X_1 - X_2)^2 X_3^2 - 2((X_1 + X_2)(X_1 X_2 + a) + 2b)X_3 \\ + ((X_1 X_2 - a)^2 - 4b(X_1 + X_2))$$

# Semaev's summation polynomials

- In general, for any  $m \geq 4$ , and  $m - 3 \geq k \geq 1$ ,  
 $f_m(X_1, \dots, X_m) =$

$$\text{Res}_X(f_{m-k}(X_1, \dots, X_{m-k-1}, X), f_{k+2}(X_{m-k}, \dots, X_m, X))$$

- Degree of  $f_m$  in each  $X_i$  is  $2^{m-2}$  for  $m \geq 3$ .
- If  $\mathbb{F}_q = \mathbb{F}_p$ , natural factor base is  
 $\{P = (x, y) \in E \text{ s.t. } x < p^{1/n}\}$
- However no known way to efficiently find such small roots  
 $x_1, \dots, x_m$  of  $f_{m+1}(x_1, \dots, x_m, x_R) = 0$  corresponding to

$$R = P_{i_1} + \dots + P_{i_m}$$

- For  $m \geq 5$  would give sub-square-root DLP algorithm

## Gaudry's insight

Let  $E : Y^2 = X^3 + aX + b$ , over a field  $\mathbb{F}_{q^n}$  with  $\text{char}(\mathbb{F}_q) > 3$

- Use a poly basis  $\{t^{n-1}, \dots, t, 1\}$  for  $\mathbb{F}_{q^n}/\mathbb{F}_q$
- Define  $\mathcal{F} = \{P = (x, y) \in E(\mathbb{F}_{q^n}) \text{ s.t. } x \in \mathbb{F}_q\}$
- Note  $|\mathcal{F}| \approx q$
- Observe that  $f_{n+1}(x_1, \dots, x_n, x_R) = 0$  now has  $n$  components:

$$f_{n+1,0} + f_{n+1,1}t + \dots + f_{n+1,n-1}t^{n-1} = 0 \in \mathbb{F}_{q^n}$$

- System of  $n$  equations over  $\mathbb{F}_q$  in  $n$  variables in  $\mathbb{F}_q$
- Solved via resultants, or Grobner basis computation

## Gaudry's insight

- Decomposition complexity  $O(\text{Poly}(2^{n(n-1)}))$
- Decomposition probability is  $1/n!$
- For fixed  $n$ ,  $q \rightarrow \infty$ , complexity is  $O(q^2)$ , rho is  $O(q^{n/2})$
- Using double large-prime variation reduces to  $O(q^{2-2/n})$
- Works for *all* curves over any extension field, even of prime extension degree.
- Details are computationally more intensive than Weil Descent.
- Subexponential attack for a large class of fields (Diem)

$$e^{O((\log q^n)^{2/3})}$$

# Algorithm complexity

**Heuristic Result 1.** *For any elliptic curve  $E(\mathbb{F}_{q^n})$ , by making  $O(q)$  queries to a Static DHP<sub>d</sub> oracle during an initial learning phase, for fixed  $n > 1$  and  $q \rightarrow \infty$ , an adversary can solve any further instance of the Static DHP<sub>d</sub> in time  $O(\text{Poly}(\log q))$ .*

# Algorithm complexity

**Heuristic Result 1.** *For any elliptic curve  $E(\mathbb{F}_{q^n})$ , by making  $O(q)$  queries to a Static DHP<sub>d</sub> oracle during an initial learning phase, for fixed  $n > 1$  and  $q \rightarrow \infty$ , an adversary can solve any further instance of the Static DHP<sub>d</sub> in time  $O(\text{Poly}(\log q))$ .*

- Can reduce the factor base à la Koblitz-Menezes:

# Algorithm complexity

**Heuristic Result 1.** *For any elliptic curve  $E(\mathbb{F}_{q^n})$ , by making  $O(q)$  queries to a Static DHP<sub>d</sub> oracle during an initial learning phase, for fixed  $n > 1$  and  $q \rightarrow \infty$ , an adversary can solve any further instance of the Static DHP<sub>d</sub> in time  $O(\text{Poly}(\log q))$ .*

- Can reduce the factor base à la Koblitz-Menezes:

**Heuristic Result 2.** *For any elliptic curve  $E(\mathbb{F}_{q^n})$ , by making  $O(q^{1-\frac{1}{n+1}})$  queries to a Static DHP<sub>d</sub> oracle during an initial learning phase, for fixed  $n > 1$  and  $q \rightarrow \infty$ , an adversary can solve any further instance of the Static DHP<sub>d</sub> in time  $\tilde{O}(q^{1-\frac{1}{n+1}})$*

# Algorithm complexity

**Heuristic Result 1.** *For any elliptic curve  $E(\mathbb{F}_{q^n})$ , by making  $O(q)$  queries to a Static DHP<sub>d</sub> oracle during an initial learning phase, for fixed  $n > 1$  and  $q \rightarrow \infty$ , an adversary can solve any further instance of the Static DHP<sub>d</sub> in time  $O(\text{Poly}(\log q))$ .*

- Can reduce the factor base à la Koblitz-Menezes:

**Heuristic Result 2.** *For any elliptic curve  $E(\mathbb{F}_{q^n})$ , by making  $O(q^{1-\frac{1}{n+1}})$  queries to a Static DHP<sub>d</sub> oracle during an initial learning phase, for fixed  $n > 1$  and  $q \rightarrow \infty$ , an adversary can solve any further instance of the Static DHP<sub>d</sub> in time  $\tilde{O}(q^{1-\frac{1}{n+1}})$*

- Can also obtain subexponential algorithm à la Diem



# The Galbraith-Lin-Scott Curves

At EUROCRYPT 2009 the use of curves defined over extension fields with degree a power of 2 were proposed.

- Exploits the existence of efficiently computable homomorphism to enable use of the GLV fast point multiplication method
- GLV: if  $\psi$  is an efficiently computable endomorphism of  $E$  then one can compute  $[n]P = [n_0]P + [n_1]\psi(P)$  with  $|n_i| \approx \sqrt{\#E}$
- Over  $\mathbb{F}_{p^2}$  method takes about 0.75 the time of the previous best methods
- Performance over  $\mathbb{F}_{p^4}$  currently uninvestigated, but subject to Gaudry's DLP attack

# The Oakley key determination protocol curves

## 'Well-Known Group' 3

Group 3 is defined over the field  $\mathbb{F}_{2^{155}} = \mathbb{F}_2[\omega]/(\omega^{155} + \omega^{62} + 1)$ , by the equation

$$Y^2 + XY = X^3 + \beta,$$

where

$$\beta = \omega^{18} + \omega^{17} + \omega^{16} + \omega^{13} + \omega^{12} + \omega^9 + \omega^8 + \omega^7 + \omega^3 + \omega^2 + \omega + 1.$$

- $\#E(\mathbb{F}_{2^{155}}) = 12 \cdot r$ , with  $r =$

3805993847215893016155463826195386266397436443

- Subject to several unsuccessful DLP attacks via Weil Descent: Jacobson/Menezes/Stein[01], Gaudry/Hess/Smart[02], Galbraith/Hess/Smart[02], Hess[03].

# The Oakley key determination protocol curves

## 'Well-Known Group' 4

Group 4 is defined over the field  $\mathbb{F}_{2^{185}} = \mathbb{F}_2[\omega]/(\omega^{185} + \omega^{69} + 1)$ , by the equation

$$Y^2 + XY = X^3 + \beta,$$

where

$$\beta = \omega^{12} + \omega^{11} + \omega^{10} + \omega^9 + \omega^7 + \omega^6 + \omega^5 + \omega^3 + 1.$$

- $\#E(\mathbb{F}_{2^{185}}) = 4 \cdot r$ , with  $r =$

12259964326927110866866776214413170562013096\  
250261263279

- DLP studied by Maurer/Menezes/Teske[01] and Menezes/Teske/Weng[04], the latter concluding that the fields  $\mathbb{F}_{2^{5l}}$  for  $l > 37$  are 'weak' while the security of ECs over  $\mathbb{F}_{2^{185}}$  is questionable

# Large prime characteristic

For each of  $n = 2, 3, 4$  and  $5$  we used curves of the form

$$E(\mathbb{F}_{p^n}) : y^2 = x^3 + ax + b,$$

for  $a$  and  $b$  randomly chosen elements of  $\mathbb{F}_{p^n}$ , such that  $\#E(\mathbb{F}_{p^n})$  was a prime of bitlength 256.

- Implemented in MAGMA (V2.16-5) run on a 3.16 GHz Intel Xeon with 32G RAM

Data for testing and decomposing points for elliptic curves over extension fields (times in s):

$n$	$\log p$	$\#f_{n+1}$	$\# \text{sym}f_{n+1}$	$T(\text{GB})$	$T(\text{roots})$
2	128	13	5	0.001	0.009
3	85.3	439	43	0.029	0.027
4	64	54777	1100	5363	3.68

# Large prime characteristic

## Upper bounds on attack time

Given data, compute  $\alpha$  such that:

$$p^{n(1-\alpha)} \cdot n! \cdot (T(\text{GB}) + T(\text{roots})) = p^\alpha \cdot T(\text{scalar})$$

# Large prime characteristic

## Upper bounds on attack time

Given data, compute  $\alpha$  such that:

$$p^{n(1-\alpha)} \cdot n! \cdot (T(\text{GB}) + T(\text{roots})) = p^\alpha \cdot T(\text{scalar})$$

Attack time estimates for our implementation (times in s):

$n$	$\alpha$	Attack time	Pollard rho
2	0.6701 (2/3)	$2^{79.8}$	$2^{111.3}$
3	0.7645 (3/4)	$2^{59.7}$	$2^{111.4}$
4	0.8730 (4/5)	$2^{50.5}$	$2^{111.4}$

## Characteristic two

For each of  $n = 2, 3, 4$  and  $5$  we used curves of the form

$$E(\mathbb{F}_{2^{ln}}) : y^2 + xy = x^3 + b, \quad (1)$$

for  $b$  a randomly chosen element of  $\mathbb{F}_{2^{ln}}$ , such that  $\#E(\mathbb{F}_{2^{ln}})$  was a four times a prime of bitlength 256.

## Characteristic two

For each of  $n = 2, 3, 4$  and  $5$  we used curves of the form

$$E(\mathbb{F}_{2^n}) : y^2 + xy = x^3 + b, \quad (1)$$

for  $b$  a randomly chosen element of  $\mathbb{F}_{2^n}$ , such that  $\#E(\mathbb{F}_{2^n})$  was a four times a prime of bitlength 256.

Data for testing and decomposing points for elliptic curves over binary extension fields and attack time estimates (times in s):

$n$	$\#f_{n+1}$	$\# \text{sym}f_{n+1}$	Time GB	$\alpha$	Attack time
2	5	3	0.000	0.6672	$2^{80.9}$
3	24	6	0.005	0.7572	$2^{60.0}$
4	729	39	247	0.8575	$2^{50.6}$
5	148300	638	N/A	N/A	N/A



## Back to Delayed Target FAPI-1 problem...

- Recall that central issue was that there is no known algorithm to invert pairing.

## Back to Delayed Target FAPI-1 problem...

- Recall that central issue was that there is no known algorithm to invert pairing.
- For binary curves defined over composite degree extension fields, natural auxiliary group comes from GHS attack

## Back to Delayed Target FAPI-1 problem...

- Recall that central issue was that there is no known algorithm to invert pairing.
- For binary curves defined over composite degree extension fields, natural auxiliary group comes from GHS attack
- **Here, one can invert the GHS homomorphism!**

## Back to Delayed Target FAPI-1 problem...

- Recall that central issue was that there is no known algorithm to invert pairing.
- For binary curves defined over composite degree extension fields, natural auxiliary group comes from GHS attack
- **Here, one can invert the GHS homomorphism!**
- Hence can mimic Delayed Target FAPI-1 problem algorithm and apply to this context

# Weil descent (Frey, Hess, Gaudry, Smart, Galbraith, Diem, Scholten,...)

- Let  $E$  be an elliptic curve over  $\mathbb{F}_{q^k}$ , with  $k > 1$ .
- Define abelian variety  $W_E$  of dimension  $k$  over  $\mathbb{F}_q$  with

$$W_E(\mathbb{F}_q) = E(\mathbb{F}_{q^k}).$$

- $W_E$  is called the Weil restriction of  $E$ .
- Try to find a curve  $H$  on  $W_E$  and map the DLOG

$$\phi : E(\mathbb{F}_{q^k}) \rightarrow \text{Jac}_H(\mathbb{F}_q).$$

- Apply index calculus to  $\text{Jac}_H(\mathbb{F}_q)$ .

## Oracle-assisted Static DHP via GHS attack

- In GHS attack elements of  $E(\mathbb{F}_{2^{ln}})[r]$  map to Jacobian of hyperelliptic curve  $H(\mathbb{F}_{2^l})$  of genus at most  $2^{n-1}$
- Let  $\mathcal{F}$  be the set of degree one divisors in  $Jac_H(\mathbb{F}_{2^l})$
- Let  $N = \#Jac_H(\mathbb{F}_{2^l})$  and  $h = N/r$
- Project each  $D \in \mathcal{F}$  into  $\text{im}(\phi)$  by multiplying by  $h$
- Compute  $\phi^{-1}(hD)$  for each  $D \in \mathcal{F}$
- Call the Static DHP<sub>d</sub> oracle on each  $\phi^{-1}(hD)$  in  $E(\mathbb{F}_{2^{ln}})$
- For a target  $X \in E(\mathbb{F}_{2^{ln}})$  take random multiples until  $\phi(aX) = \sum D_i \in \mathcal{F}$
- Then assuming  $(h, r) = 1$  one computes

$$\delta(X) = (a^{-1} \bmod r)(h^{-1} \bmod r) \sum \delta(\phi^{-1}(hD_i))$$

## GHS for 'Well-Known Group' 3

We have  $\phi : E(\mathbb{F}_{2^{155}})[r] \longrightarrow \text{Jac}_H(\mathbb{F}_{2^{31}})$  for hyperelliptic

$$H : Y^2 + h(X) \cdot Y = f(X),$$

with  $\mathbb{F}_{2^{31}} = \mathbb{F}_2[\omega]/(\omega^{31} + \omega^3 + 1)$  and

$$\begin{aligned}h(X) &= 289804524X^{16} + 607247628X^8 + 1798965180X^4 \\ &+ 1103766465X^2 + 742287012X, \\ f(X) &= 505223067X^{33} + 1000507042X^{17} + 1992775259X^{16} \\ &+ 1146351457X^9 + 1078048302X^8 + 284388091X^5 \\ &+ 518998412X^4 + 1875045691X^3 + 2001664187X^2 \\ &+ 1973705837X,\end{aligned}$$

and  $\text{genus}(H) = 16 = 2^{155/31-1}$ .

## Static DHP for 'Well-Known Group' 3 via GHS

- Using Florian's LMS J. Comput. Math paper (or a magma computation), one finds  $N = \#\text{Jac}_H(\mathbb{F}_{2^{31}})$  which has bitlength 497
- Furthermore  $(N/r, r) = 1$  and so attack can proceed
- Using Victor Shoup's Number Theory Library on a 3.16GHz Intel Xeon, testing 1-smoothness of a random multiple of  $\phi(P)$  takes  $\approx 0.690ms$
- Other basic cost is a point addition in the Jacobian; Jacobson estimates this to be  $< 1/2.3$  the cost of smoothness test using NUCOMP
- Hence expected time to find a relation using a single processor is  $\approx 650$  years.



## GHS for 'Well-Known Group' 4

We have  $\phi : E(\mathbb{F}_{2^{185}})[r] \longrightarrow \text{Jac}_H(\mathbb{F}_{2^{37}})$  for hyperelliptic

$$H : Y^2 + h(X) \cdot Y = f(X),$$

with  $\mathbb{F}_{2^{37}} = \mathbb{F}_2[\omega]/(\omega^{37} + \omega^9 + \omega^2 + \omega + 1)$  and

$$\begin{aligned} h(X) &= 73994877348X^{16} + 113350789030X^8 + 86827085475X^4 \\ &+ 21964938327X^2 + 125543309305X, \\ f(X) &= 49045248530X^{33} + 40737336296X^{17} + 45140903646X^{16} \\ &+ 120039047741X^9 + 105120752497X^8 + 72787224919X^5 \\ &+ 25040887869X^4 + 72047225547X^3 + 94586877616X^2 \\ &+ 68639477599X, \end{aligned}$$

and  $\text{genus}(H) = 16 = 2^{185/37-1}$ .

## Static DHP for 'Well-Known Group' 4 via GHS

- $N = \#\text{Jac}_H(\mathbb{F}_{2^{37}})$  has bitlength 592
- Furthermore  $(N/r, r) = 1$  and so attack can proceed
- Using Victor Shoup's Number Theory Library on a 3.16GHz Intel Xeon, testing 1-smoothness of a random multiple of  $\phi(P)$  takes  $\approx 0.854ms$
- Other basic cost is a point addition in the Jacobian; Jacobson estimates this to be  $\approx 1/2.3$  the cost of smoothness test using NUCOMP
- Hence expected time to find a relation using a single processor is  $\approx 810$  years.

## Static DHP for $E(\mathbb{F}_{2^{ln}})$ via GHS

Components of learning phase:

- Construct factor base  $\mathcal{F}$  of degree 1 divisors:  $\approx 2^{l-1}$  such divisors ignoring negatives
- Map each  $D \in \mathcal{F}$  to an element of  $\text{im}(\phi)$  via multiplication by  $h = \#\text{Jac}_H(\mathbb{F}_{2^l})/r \approx 2^{l((2^{n-1})!-n)}$
- Compute  $\phi^{-1}(hD)$  for each  $D \in \mathcal{F}$
- Call the Static DHP $_d$  oracle on each  $\phi^{-1}(hD)$  in  $E(\mathbb{F}_{2^{ln}})$

Expected cost of relation find:

- Cost of each smoothness test  $\approx (128l - 288) \mathbb{F}_{2^l}$  multiplications
- Hence total cost is  $\approx (2^{n-1})! \cdot (128l - 288) \mathbb{F}_{2^l}$  multiplications

## Static DHP for $E(\mathbb{F}_{2^l n})$ via GHS

Consider asymptotics for fixed  $n$  and  $l \rightarrow \infty$ . Write  $g = 2^{n-1}$ .

- For  $2^l > g!$  the dominant cost is the oracle calls
- Hence should reduce  $\mathcal{F}$  to balance the two stages
- Let  $q = 2^l$  and let  $|\mathcal{F}_s| = q^\alpha$  with  $0 < \alpha \leq 1$
- Probability that a random point decomposes over  $\mathcal{F}_s$  is  $q^{g(\alpha-1)}/g!$

Solving  $g! \cdot q^{g(1-\alpha)} = q^\alpha$  gives  $\alpha = \frac{g + \log_q g!}{g+1}$  and so complexity of algorithm is

$$\tilde{O}(q^{1-\frac{1}{g+1}}).$$

- This is the square-root of the balanced DLP algorithm complexity for fixed genus (Gaudry/Harley)

## Comparison with the Gaudry/Semaev-based method

- For fixed  $n$  and increasing  $q$  first algorithm is asymptotically faster:  $\tilde{O}(q^{1-\frac{1}{n+1}})$  vs  $\tilde{O}(q^{1-\frac{1}{g+1}})$
- In practice, factorisation is much easier than a decomposition — have a trade-off between factorisation probability and ease of factorisation — so may even be better for  $n = 2, 3, 4$ , as well as 5
- Method is really tailored for when Gaudry/Semaev decompositions are impractical
- Limitation: details are only clear in characteristic 2

## Conclusions/Questions

- Some problems occurring in security proofs are easier than DLP when index calculus applies

## Conclusions/Questions

- Some problems occurring in security proofs are easier than DLP when index calculus applies
- Elliptic curves defined over extension fields may be unsuitable in some scenarios

## Conclusions/Questions

- Some problems occurring in security proofs are easier than DLP when index calculus applies
- Elliptic curves defined over extension fields may be unsuitable in some scenarios
- Oracle-assisted pairing inversion seems interesting



## Conclusions/Questions

- Some problems occurring in security proofs are easier than DLP when index calculus applies
- Elliptic curves defined over extension fields may be unsuitable in some scenarios
- Oracle-assisted pairing inversion seems interesting
- Further applications of Weil descent yet to be revealed?