Asiacrypt 2021

Cryptanalysis of an Oblivious PRF from Supersingular Isogenies

Andrea Basso, Péter Kutas, Simon-Philipp Merz, Christophe Petit and Antonio Sanso

This Talk

- What is an OPRF
- The OPRF from isogenies
- A polytime attack
- A subexponential attack
- Implementation results
- Need for a trusted setup

Oblivious Pseudorandom Functions (OPRF)

An OPRF is a two-party protocol to evaluate a PRF $f(k, m)$ where:

- the client learns $f(k, m)$, one evaluation of a PRF on a chosen input
- the server learns nothing
- if the OPRF is *verifable*, the server always uses the same key k

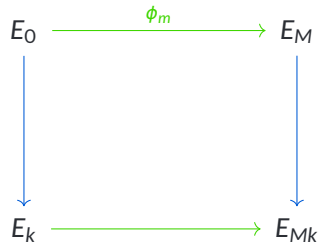
Applications:

- password-authenticated key exchanges,
- private-set interesection,
- privacy-preserving CAPTCHA

Oblivious Pseudorandom Functions from Isogenies [BKW20]

Client

Server



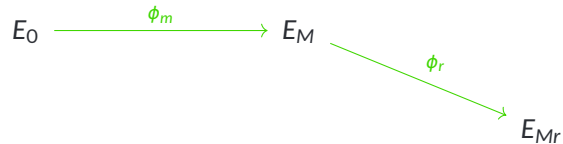
Oblivious Pseudorandom Functions from Isogenies [BKW20]

Client

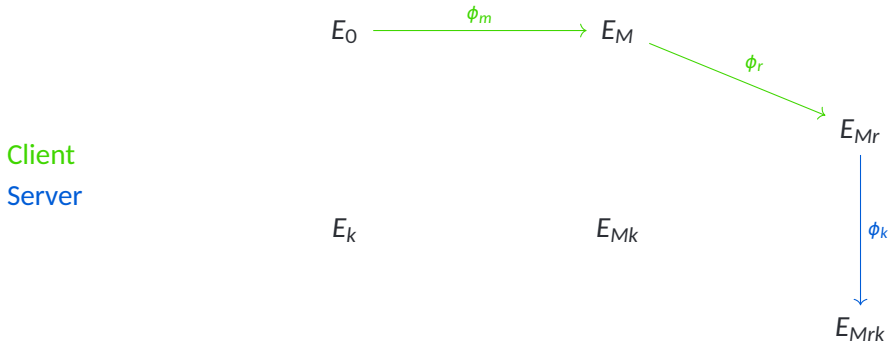
Server

E_k E_{Mk}

Oblivious Pseudorandom Functions from Isogenies [BKW20]



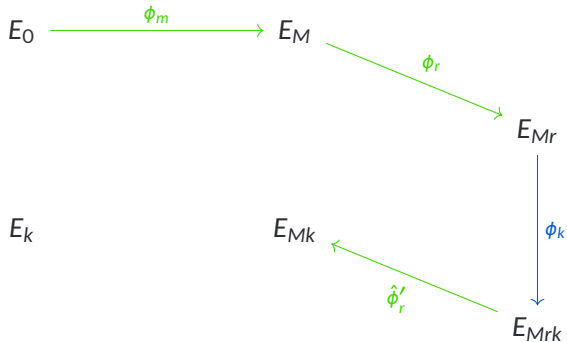
Oblivious Pseudorandom Functions from Isogenies [BKW20]



Oblivious Pseudorandom Functions from Isogenies [BKW20]

Client

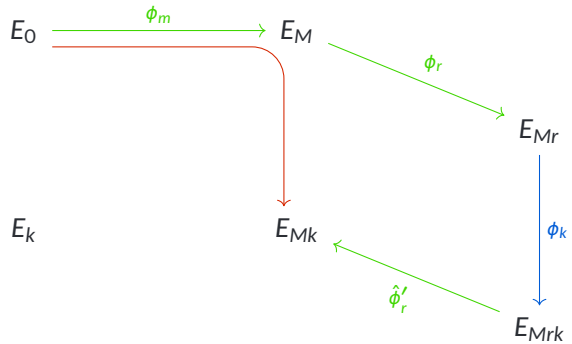
Server



Oblivious Pseudorandom Functions from Isogenies [BKW20]

Client

Server



$$f(k, m) = H(m, j(E_{Mk}), \text{pk})$$

The ‘One-More’ Assumption

The ‘one-more’ assumption states that an attacker cannot evaluate the OPRF even after multiple queries

The ‘One-More’ Assumption

The ‘one-more’ assumption states that an attacker cannot evaluate the OPRF even after multiple queries

The game:

- Attacker sends a point M_0 on E_0

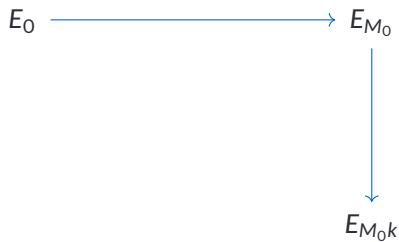
E_0

The ‘One-More’ Assumption

The ‘one-more’ assumption states that an attacker cannot evaluate the OPRF even after multiple queries

The game:

- Attacker sends a point M_0 on E_0
- Challenger replies with $E_0/\langle M_0, K\rangle$

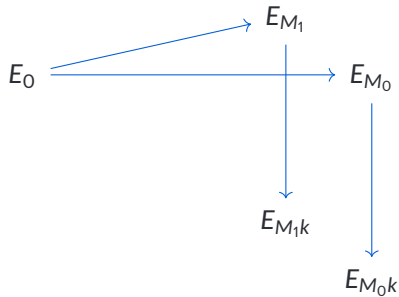


The ‘One-More’ Assumption

The ‘one-more’ assumption states that an attacker cannot evaluate the OPRF even after multiple queries

The game:

- Attacker sends a point M_0 on E_0
- Challenger replies with $E_0/\langle M_0, K\rangle$
- Repeat multiple times

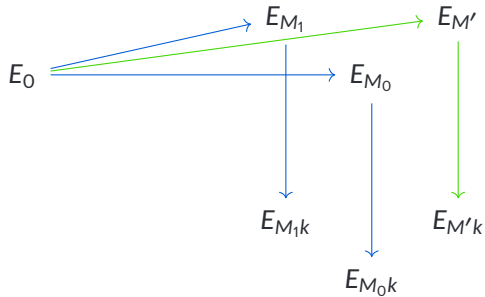


The ‘One-More’ Assumption

The ‘one-more’ assumption states that an attacker cannot evaluate the OPRF even after multiple queries

The game:

- Attacker sends a point M_0 on E_0
- Challenger replies with $E_0/\langle M_0, K\rangle$
- Repeat multiple times
- Attacker outputs $E_0/\langle M', K\rangle$ for M' chosen by the challenger



Attack Strategy

Using multiple queries, we want to:

1. Find E_k and $\langle \phi_k(M) \rangle$ for some point $M \in E_0[2^n]$
2. Combine multiple points to obtain $\phi_k(E_0[2^n])$

The attack then becomes:

1. For any point $P \in E_0[2^n]$, compute $\langle \phi_k(P) \rangle$
2. Output $E_k / \langle \phi_k(P) \rangle = E_0 / \langle P, K \rangle$

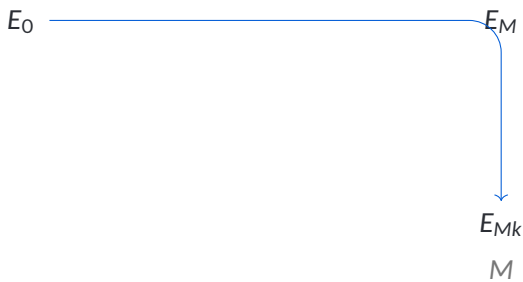
A Polytime Attack

Recovering points on E_k

E_0

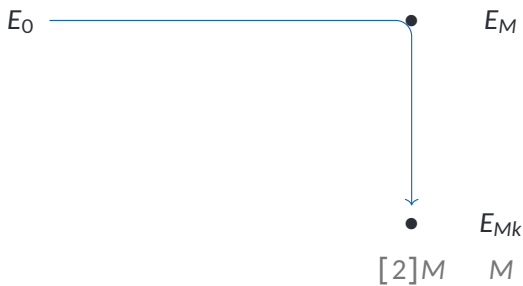
A Polytime Attack

Recovering points on E_k



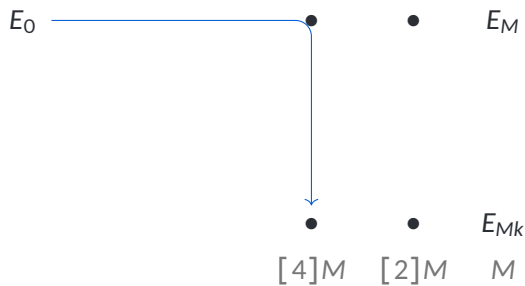
A Polytime Attack

Recovering points on E_k



A Polytime Attack

Recovering points on E_k

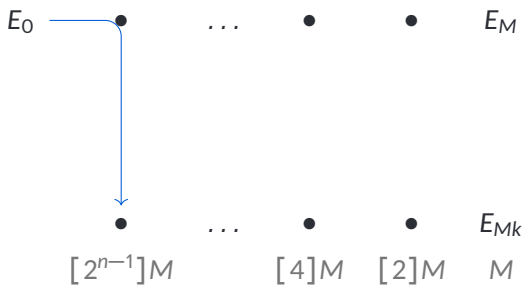


A Polytime Attack

Recovering points on E_k

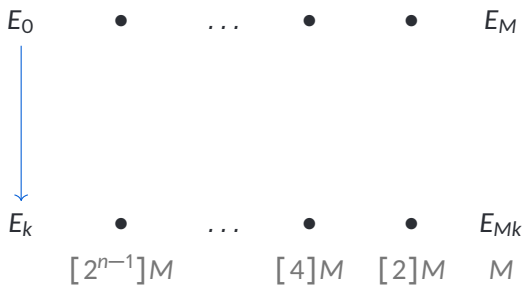
A Polytime Attack

Recovering points on E_k



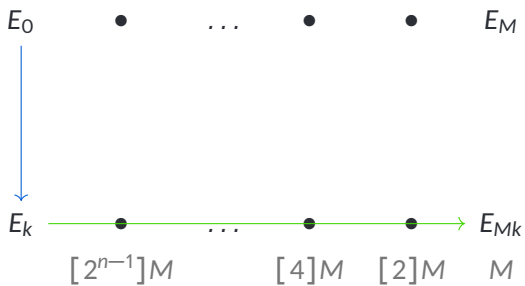
A Polytime Attack

Recovering points on E_k



A Polytime Attack

Recovering points on E_k



A Polytime Attack

Recovering points on E_k

$$\ker \phi = \langle \phi_K(M) \rangle$$

A Polytime Attack

Combining the points

Given M on $E_0[2^n]$, we can recover $\langle \phi_K(M) \rangle$

A Polytime Attack

Combining the points

Given M on $E_0[2^n]$, we can recover $\langle \phi_K(M) \rangle$ $\Rightarrow$ we can recover $[\alpha]\phi_K(M)$

A Polytime Attack

Combining the points

Given M on $E_0[2^n]$, we can recover $\langle \phi_K(M) \rangle \quad \Rightarrow \quad$ we can recover $[\alpha]\phi_K(M)$

We query on M, N, $M+N$ and obtain

$$M' = [\alpha]\phi_K(M)$$
$$N' = [\beta]\phi_K(N)$$
$$R' = [\gamma]\phi_K(M+N) = [a]M' + [b]N'$$

A Polytime Attack

Combining the points

Given M on $E_0[2^n]$, we can recover $\langle \phi_K(M) \rangle \quad \Rightarrow \quad$ we can recover $[\alpha]\phi_K(M)$

We query on M, N, $M+N$ and obtain

$$\left.\begin{aligned} M' &= [\alpha]\phi_K(M) \\ N' &= [\beta]\phi_K(N) \\ R' &= [\gamma]\phi_K(M+N) = [a]M' + [b]N' \end{aligned}\right\} \Rightarrow \frac{\alpha}{\beta} = \frac{b}{a}$$

A Polytime Attack

Combining the points

Given M on $E_0[2^n]$, we can recover $\langle \phi_K(M) \rangle$ $\Rightarrow$ we can recover $[\alpha]\phi_K(M)$

We query on M, N, $M+N$ and obtain

$$\left.\begin{aligned} M' &= [\alpha]\phi_K(M) \\ N' &= [\beta]\phi_K(N) \\ R' &= [\gamma]\phi_K(M+N) = [a]M' + [b]N' \end{aligned}\right\} \Rightarrow \frac{\alpha}{\beta} = \frac{b}{a}$$

Breaking the assumption

Given any $P = [x]M + [y]N$, we can compute $\langle \phi_K(P) \rangle = \langle [x]M' + [y]\frac{\alpha}{\beta}N' \rangle$

A Polytime Attack

Results

- $O(\lambda)$ queries recover $\langle \phi_K(M) \rangle$ for any M in $E_0[2^n]$
- With three subgroups, we can compute $\langle \phi_K(P) \rangle$ for any P without further interactions
- This breaks the ‘one-more’ assumption

A Polytime Attack

Results

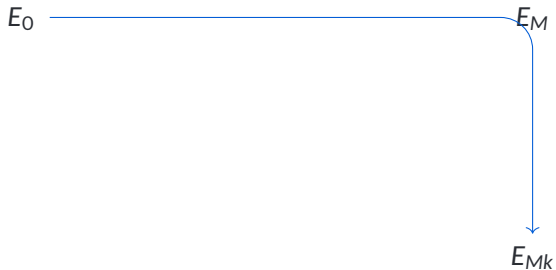
- $O(\lambda)$ queries recover $\langle \phi_K(M) \rangle$ for any M in $E_0[2^n]$
- With three subgroups, we can compute $\langle \phi_K(P) \rangle$ for any P without further interactions
- This breaks the ‘one-more’ assumption

But

- It is easy to check that query points have full order
- The OPRF protocol uses ZKP that also guarantee full order

A Subexponential Attack

Using full-order queries



A Subexponential Attack

Using full-order queries

E_0

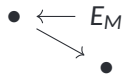
$\bullet \longleftarrow E_M$

E_{Mk}

A Subexponential Attack

Using full-order queries

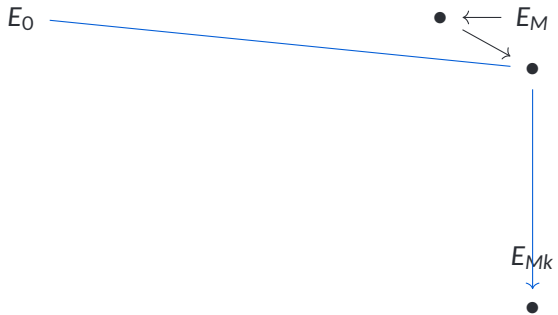
E_0



E_{Mk}

A Subexponential Attack

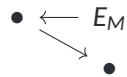
Using full-order queries



A Subexponential Attack

Using full-order queries

E_0



A Subexponential Attack

Using full-order queries

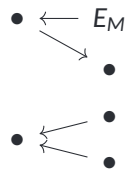
A Subexponential Attack

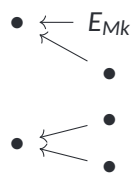
Using full-order queries

A Subexponential Attack

Using full-order queries

E_0

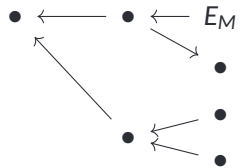


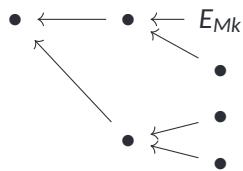


A Subexponential Attack

Using full-order queries

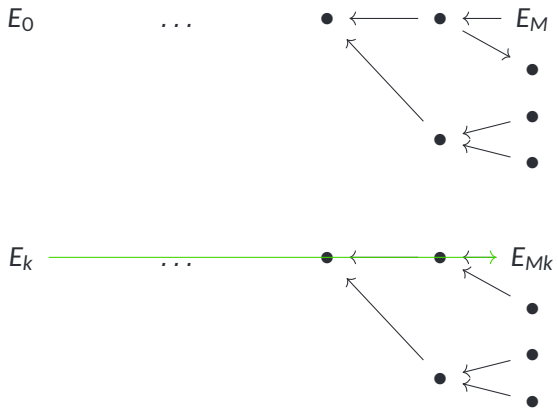
E_0





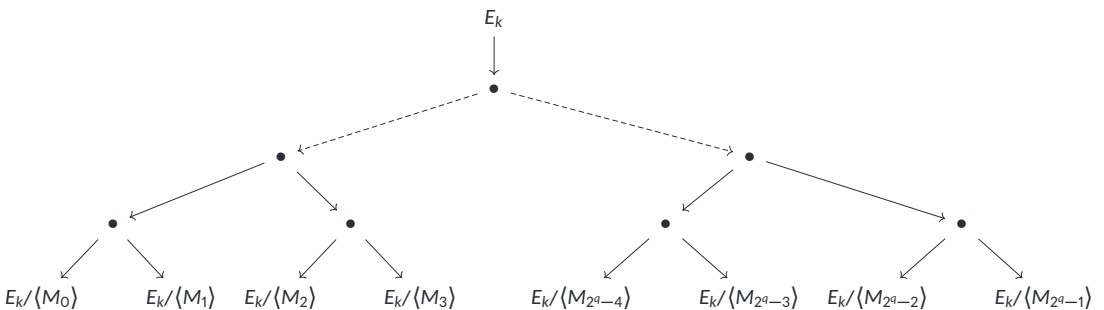
A Subexponential Attack

Using full-order queries



A Subexponential Attack

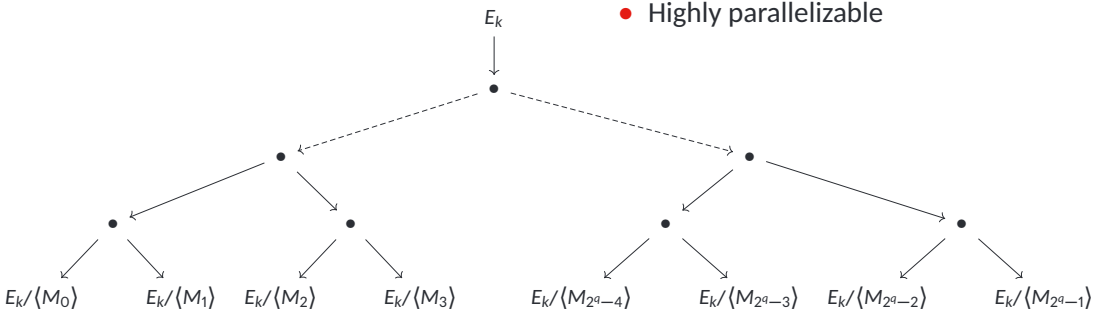
Building a tree



A Subexponential Attack

Building a tree

- Queries/complexity trade-offs
  - $O(2\lambda/3)$ complexity with 2 queries
- Highly parallelizable



A Subexponential Attack

The full attack:

- Use the binary tree to recover points on E_k
- Second part of the attack same as polytime attack
- Subexponential complexity for balanced trade-offs

Countermeasures:

- No obvious countermeasures
- Increase the parameter size? $\Rightarrow$ very large degrees
- New efficient solutions?

Implementation Results

Parameters				MITM		Running Time
$\log p$	λ	n	q	Distance	Memory (kB)	(s)
112	8	20	3	8	3.5	15
216	16	40	6	10	13.8	212 (3.53 m)
413	32	80	8	16	211.4	1,371 (22.85 m)
859	67	169	11	26	14,073	163,869 (1.89 d)
1,614	128	320	18	40	3,384,803	*174,709,440 (5.54 y)*

Available at https://github.com/isogenists/isogeny-OPRF

The Starting Curve

Who chooses E_0?

- The client
- A third-party
- The server
- Known curve ($j(E_0) = 1728$)
- Trusted setup

The Starting Curve

Who chooses E_0?

- The client
- A third-party

} can backdoor E_0 $\Rightarrow$ key-recovery attack on the server

- The server
- Known curve ($j(E_0) = 1728$)
- Trusted setup

The Starting Curve

Who chooses E_0?

- The client
- A third-party

} can backdoor E_0 $\Rightarrow$ key-recovery attack on the server

- The server
- Known curve ($j(E_0) = 1728$)
- Trusted setup

} breaks the *Supersingular Isogeny Collision* assumption

The Starting Curve

Who chooses E_0?

- The client
- A third-party

} can backdoor E_0 $\Rightarrow$ key-recovery attack on the server

- The server
- Known curve ($j(E_0) = 1728$)
- **Trusted setup**

} breaks the *Supersingular Isogeny Collision* assumption

Conclusion

- Two attacks on the ‘one-more’ assumption
- A subexponential attack on the OPRF protocol
- An implementation that shows the feasibility of the attack
- Need for a trusted setup

Paper available at `https://ia.cr/2021/706`

Future work

- Improving the attack complexity
- New efficient countermeasures
- Further study the trusted setup requirement

References I

[BKW20] Dan Boneh, Dmitry Kogan, and Katharine Woo.
Oblivious pseudorandom functions from isogenies.
In *Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part II*, pages 520–550, 2020.