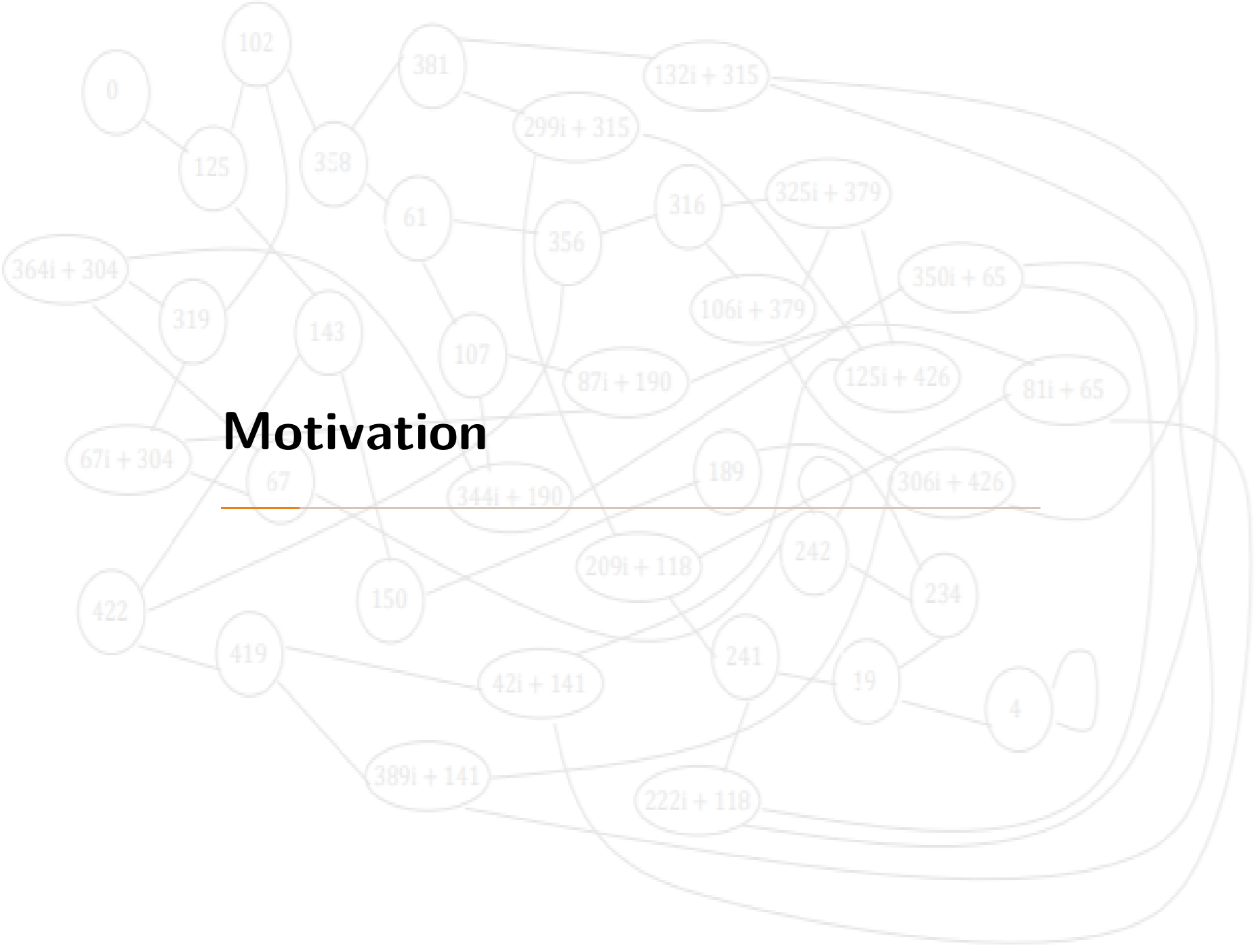
git commit -m “isogenies”

Bruno Sterner, b.sterner@surrey.ac.uk

Talk at the Isogeny Club

Disclaimer

Depsite the talk title I neither endorse nor not endorse the use of git!


102
0
381
132i + 315
299i + 315
125
358
61
316
325i + 379
356
364i + 304
350i + 65
106i + 379
319
143
107
87i + 190
125i + 426
81i + 65
67i + 304
67
189
306i + 426
344i + 190
242
209i + 118
150
234
422
419
241
42i + 141
19
4
389i + 141
222i + 118

Motivation

My journey

If we have isogeny analogues of Diffie Hellman then why don't we have an isogeny analogue of the Pedersen commitment?

Because it's not quite so simple! It's not a direct analogue but there is a solution. In addition, some interesting properties of isogeny graphs are explored

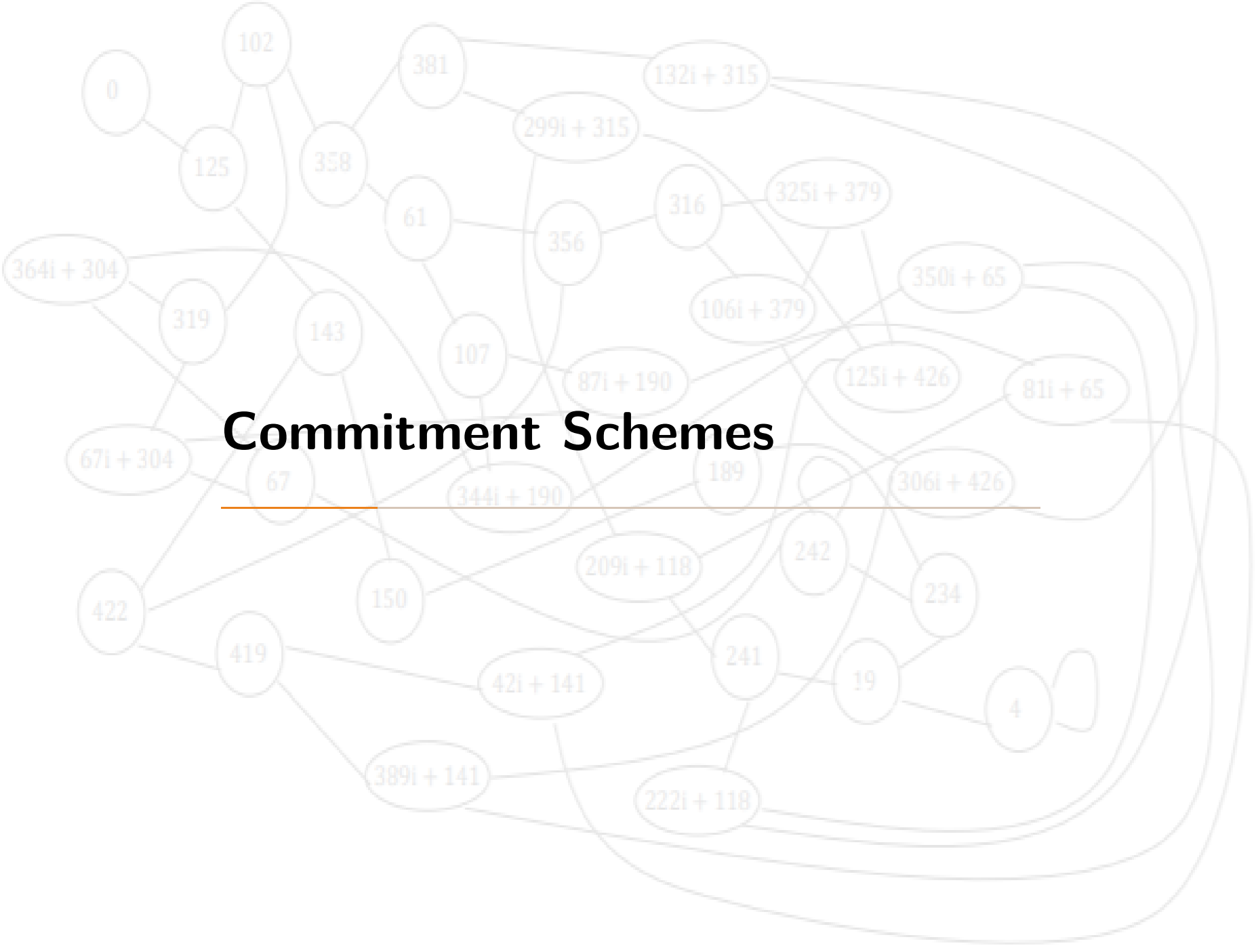
Outline

Commitment Schemes

Isogeny-Based Cryptography

Supersingular Isogeny Graphs

git commit -m ”isogenies”



Commitment Schemes

What are Commitment Schemes?

A **commitment scheme** is a two stage protocol:

1) Alice commits to a message while keeping it hidden from Bob
2) Alice reveals the message and Bob can verify that this is the correct message

	Alice		**Bob**
Commit:	m, r $c = \text{Commit}(m, r)$	$\xrightarrow{\quad c \quad}$	Save c
Open:		$\xrightarrow{\quad m, r, c \quad}$	Accept if $c = \text{Commit}(m, r)$

Required Properties for Commitment Schemes

There are two important properties needed to make commitments cryptographically useful:

Hiding: No information about the message can be attained from the commitment;

Binding: It should not be possible to replicate the same commitment with a different message.

Example 1: Hash Functions

Setup

Fix a cryptographic hash function $H : \{0,1\}^* \to \{0,1\}^\kappa$

Commit: $m \in \{0,1\}^*$, $r \in_R \{0,1\}^*$

$$c = H(m||r).$$

Open: Output m, r, c

Information-Theoretic Hiding

Assuming H is modelled as a random oracle, resulting commitment appears to be something completely random

Computational Binding

Binding reduces down to collision resistance of H

Example 2: Pedersen Commitment

Setup

Fix a finite cyclic group $G = \langle g \rangle$ of prime order p with a hard discrete logarithm problem and a random $h \in G$

Commit: $m \in \mathbb{Z}/p\mathbb{Z}$, $r \in_R \mathbb{Z}/p\mathbb{Z}$

$$c = g^m h^r.$$

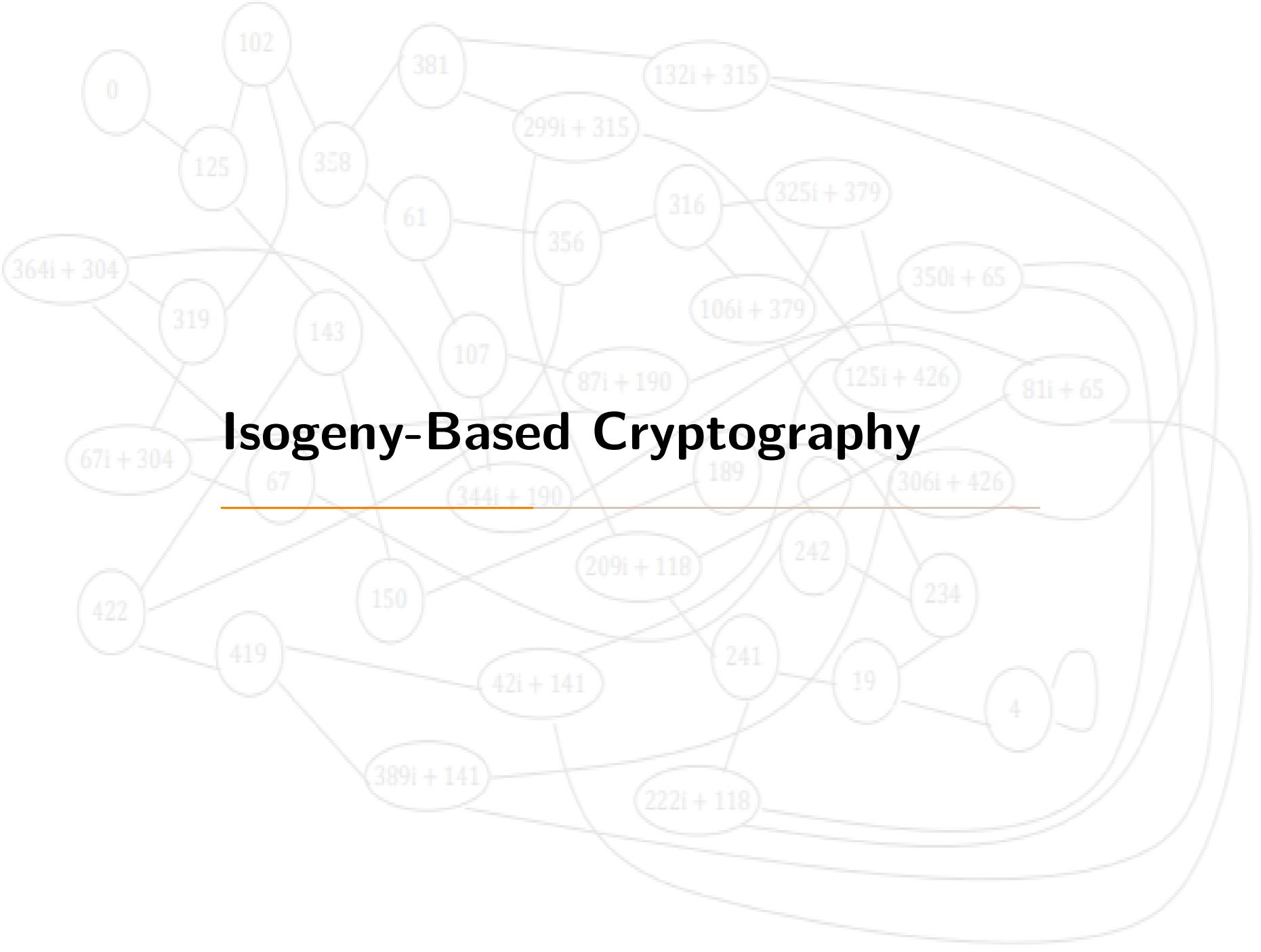
Open: Output m, r, c

Perfect Hiding

Since r was chosen randomly, any message m' could give you a valid commitment c

Computational Binding

Binding reduces down to solving the discrete logarithm of h in base g



Isogeny-Based Cryptography

Background on Isogenies

Isogeny:

$$\phi : E \to E'$$

The degree of a "*separable*" isogeny is $\deg(\phi) = \ker(\phi)$

ℓ-isogeny: An isogeny of degree ℓ

Example

A 2-isogeny between $E : y^2 = x^3 + x$ and $E' : y^2 = x^3 - 4x$:

$$\phi : (x, y) \mapsto \left(\frac{x^2 + 1}{x}, y \frac{x^2 - 1}{x^2} \right).$$

Endomorphism Rings

Endomorphism: An isogeny with $E' = E$

Endomorphism Ring: $\mathrm{End}(E) = \{\text{endomorphisms of E}\}$

We always have $\mathbb{Z} \hookrightarrow \mathrm{End}(E)$ by scalar multiplications

If $E/\mathbb{F}_{p^k}$, then $\mathrm{End}(E)$ contains the Frobenius endomorphism π

Supersingular Elliptic Curves:

$$\mathrm{End}(E) \cong \text{maximal order in a quaternion algebra}$$

Example

When $p = 3 \mod 4$, the curve $E : y^2 = x^3 + x$ is supersingular with

$$\mathrm{End}(E) = \langle 1, \iota, \frac{\iota + \pi}{2}, \frac{1 + \iota\pi}{2} \rangle$$

Isogeny Graphs

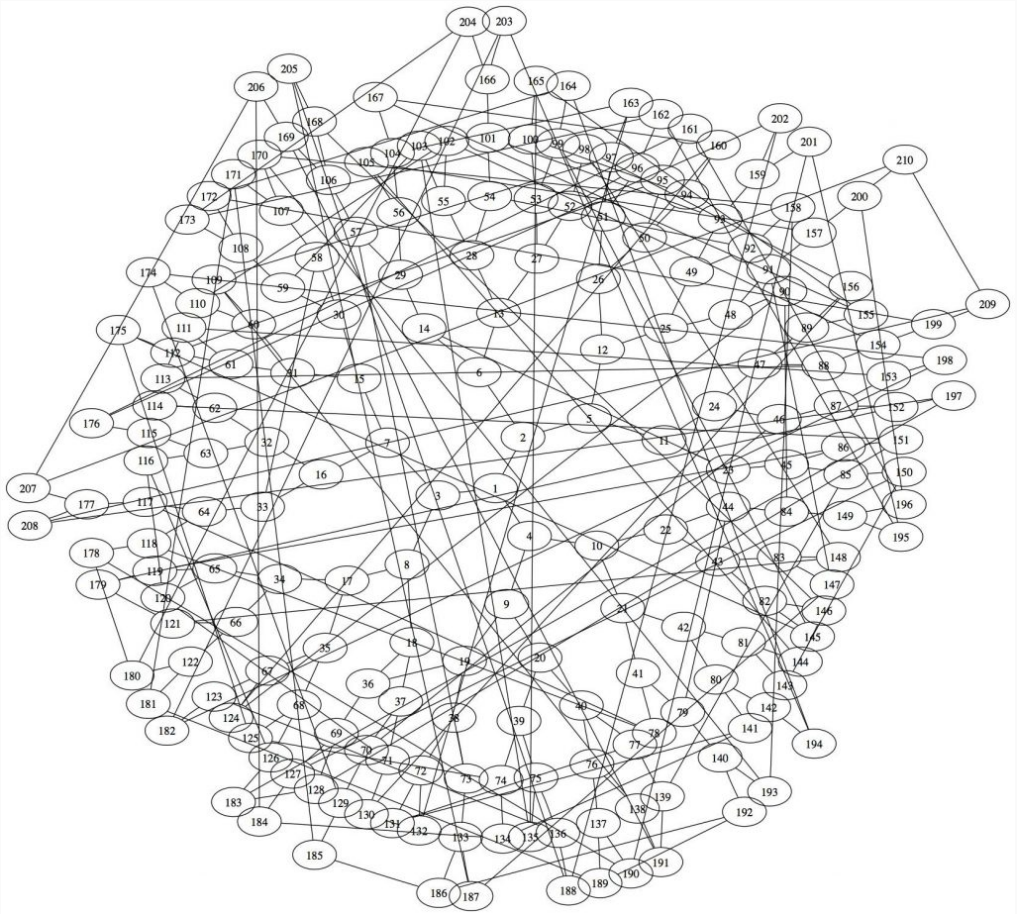
Supersingular ℓ-isogeny graph: Directed graph consisting of vertices and edges

$$\mathcal{V} = \{j\text{-invariants of supersingular elliptic curves}\}, \quad \mathcal{E} = \{\ell\text{-isogenies}\}$$

Properties include:

- Connected and $(\ell + 1)$-regular
- Vertices can be labelled by an element in $\mathbb{F}_{p^2}$
- Number of vertices is around $p/12$
- Ramanujan: optimal expansion properties

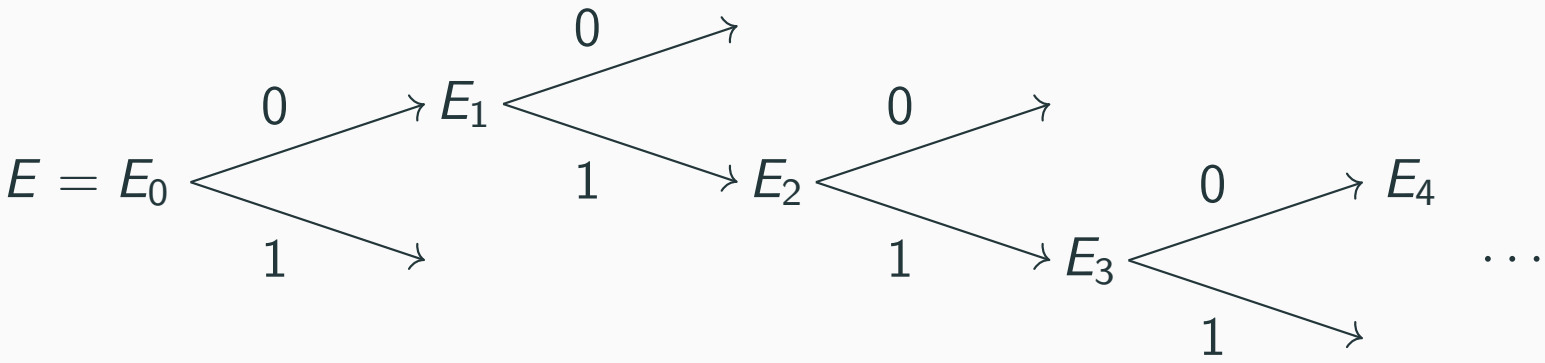
Isogeny Graph



Source: "Where cryptography and quantum computing intersect", Microsoft Research Blog, May 8th 2017

CGL Hash Function

Fix a supersingular elliptic curve $E/\mathbb{F}_{p^2}$ and walk in the 2-isogeny graph



The hash of the binary string $[0, 1, 1, 0]$ is $j(E_4)$

Underlying security relies on the hardness of the **Isogeny Path Problem** which is provably equivalent to the **Endomorphism Ring Problem**

Knowledge of Endomorphism Ring in CGL

If $\mathrm{End}(E)$ is known then one can break the second preimage resistance property of the hash function

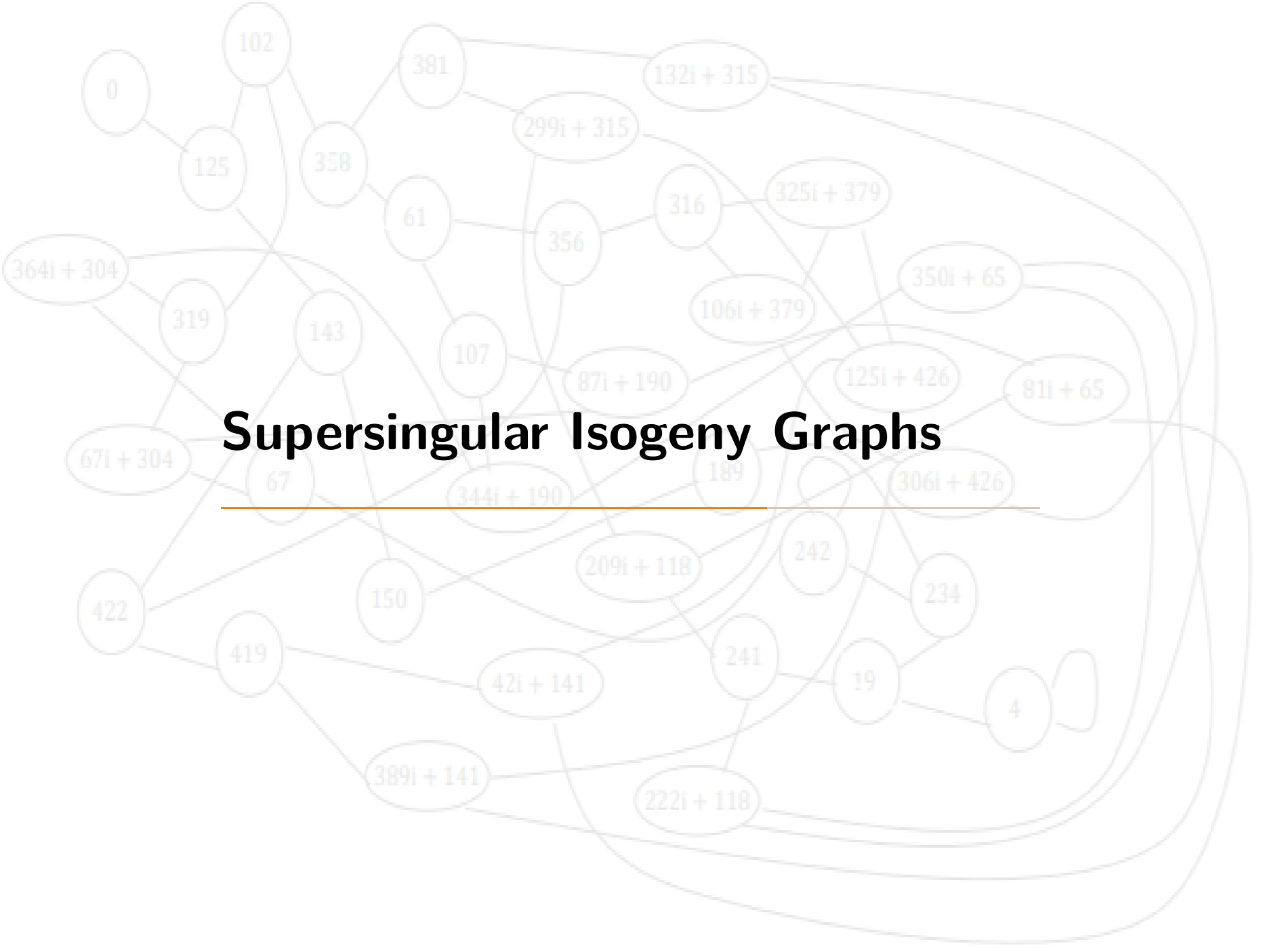
This rules out choosing explicit supersingular curves: e.g. $E : y^2 = x^3 + x$

Hard Supersingular Curve

Find a supersingular curve, E, such that its endomorphism ring, $\mathrm{End}(E)$, is completely unknown.

At least without a trusted setup, this remains an open problem

Such hard curves have applications to other protocols including VDF's



Supersingular Isogeny Graphs

Non-Bactracking Walks on Regular Graphs

A lot is known about walking on certain graphs (in the generic sense)

For what we need later on, we need to restrict ourselves to **non-backtracking** walks

Informal Lemma

On a connected d-regular graph G, there is a positive integer k_G such that any two vertices in G are connected by a non-backtracking path of any length $\geq k_G$.

If k_G is chosen minimally, then we call it the **mixing constant** of G

Bounds for the Mixing Constant

Lower Bound:

If N is the number of vertices in G, then

$$k_G \geq \log_{d-1}(N) - \log_{d-1}(d) + 1$$

Upper Bound:

Heuristically, if G behaves like a random regular graph then one can show that

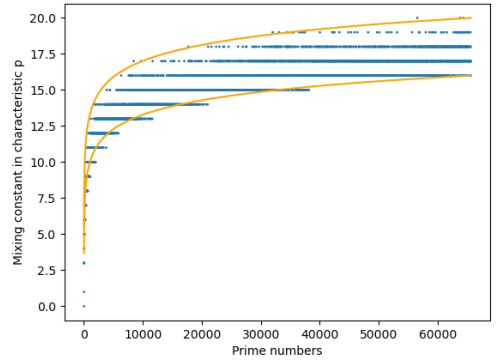
$$k_G \leq 4 \log_{d-1}(dN) + 4$$

This is in some sense a worst case bound among all regular graphs

Restriction to Supersingular Isogeny Graphs

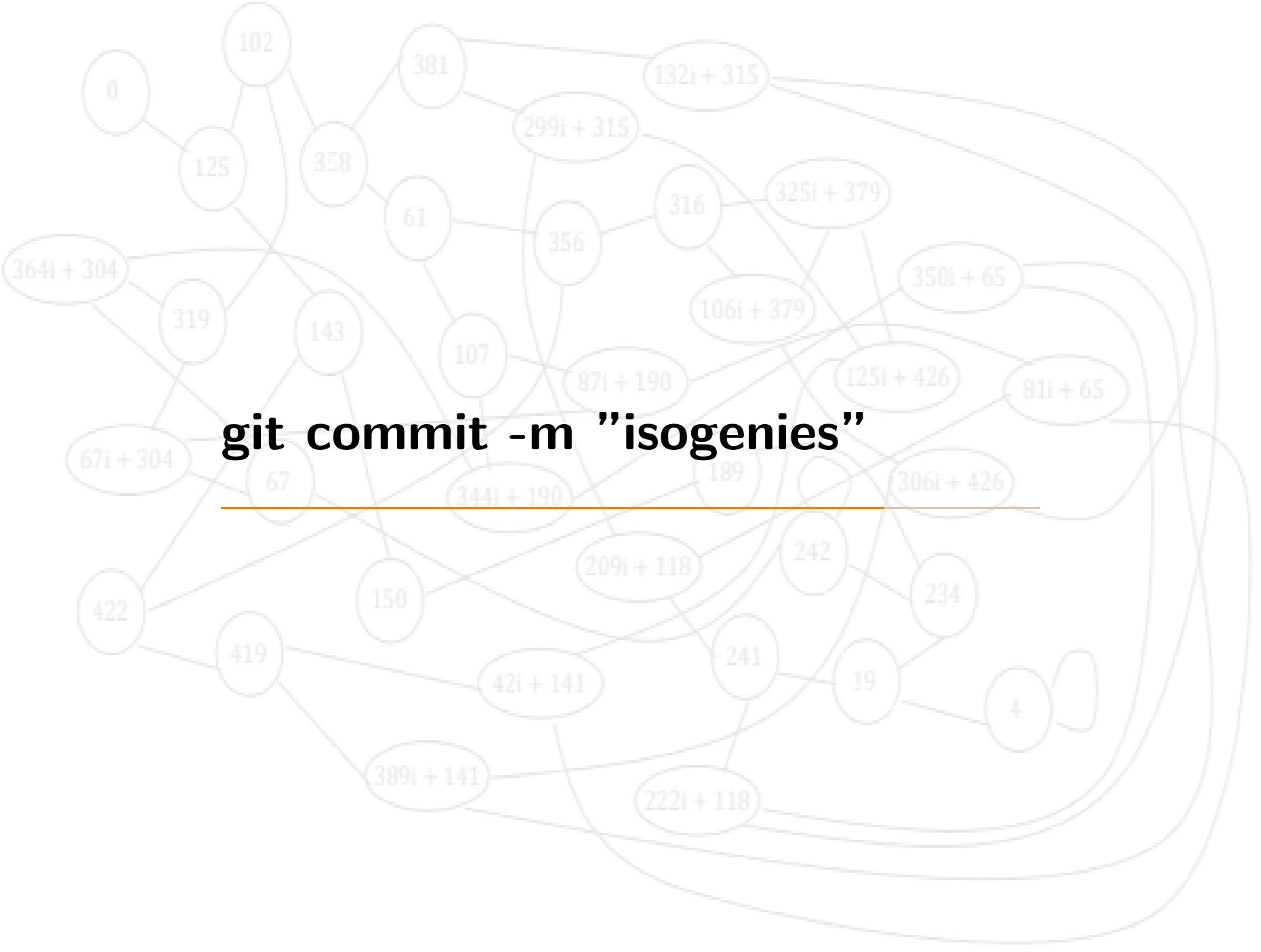
Let $k_{\ell,p}$ be the mixing contant for supersingular isogeny graphs

Following experimental results we conjecture that $k_{\ell,p} \leq \log_\ell(p) + \log_\ell(\log_\ell(p)) + O(1)$



The addition of $\log_\ell(\log_\ell(p))$ here is unique among graph theoretic constants of these isogeny graphs

Small remark: The conjectured upper bound along with the lower bound appear to be sharp (as showcased in the figure)



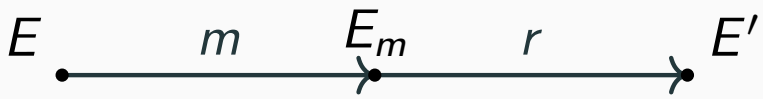
git commit -m ”isogenies”

Our Commitment Scheme Construction

Setup

Choose a hard supersingular elliptic curve $E/\mathbb{F}_{p^2}$, two random edges incident to $j(E)$ in the 2-isogeny graph and a positive integer k

Commit: Given $m \in \{0,1\}^k$, choose $r \in_R \{0,1\}^k$ and go on a walk in the 2-isogeny graph:



Then return $c := j(E')$

Open: Upon receiving m and r, recompute E' and return $c == j(E')$

Information-Theoretic Hiding of our Commitment Scheme

Theorem

Let $k_{2,p}$ be the mixing constant of the supersingular 2-isogeny graph. Then for any $k \geq k_{2,p}$ our commitment scheme is information-theoretically hiding.

Sketch of the proof.

For two messages m, m' the choice of k guarantees the existence of length k paths in the 2-isogeny graph between E_m, E' and $E_{m'}, E'$. Distinguishing between these cases can be done with negligible probability. □

No random oracles are needed in the security proof which is the main difference between this and the generic approach using hash functions

Computational Binding of our Commitment Scheme

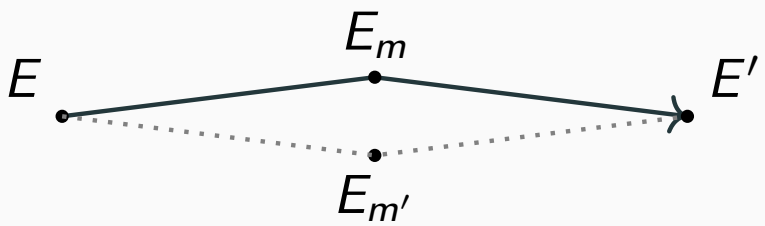
Supersingular Smooth Endomorphism Problem

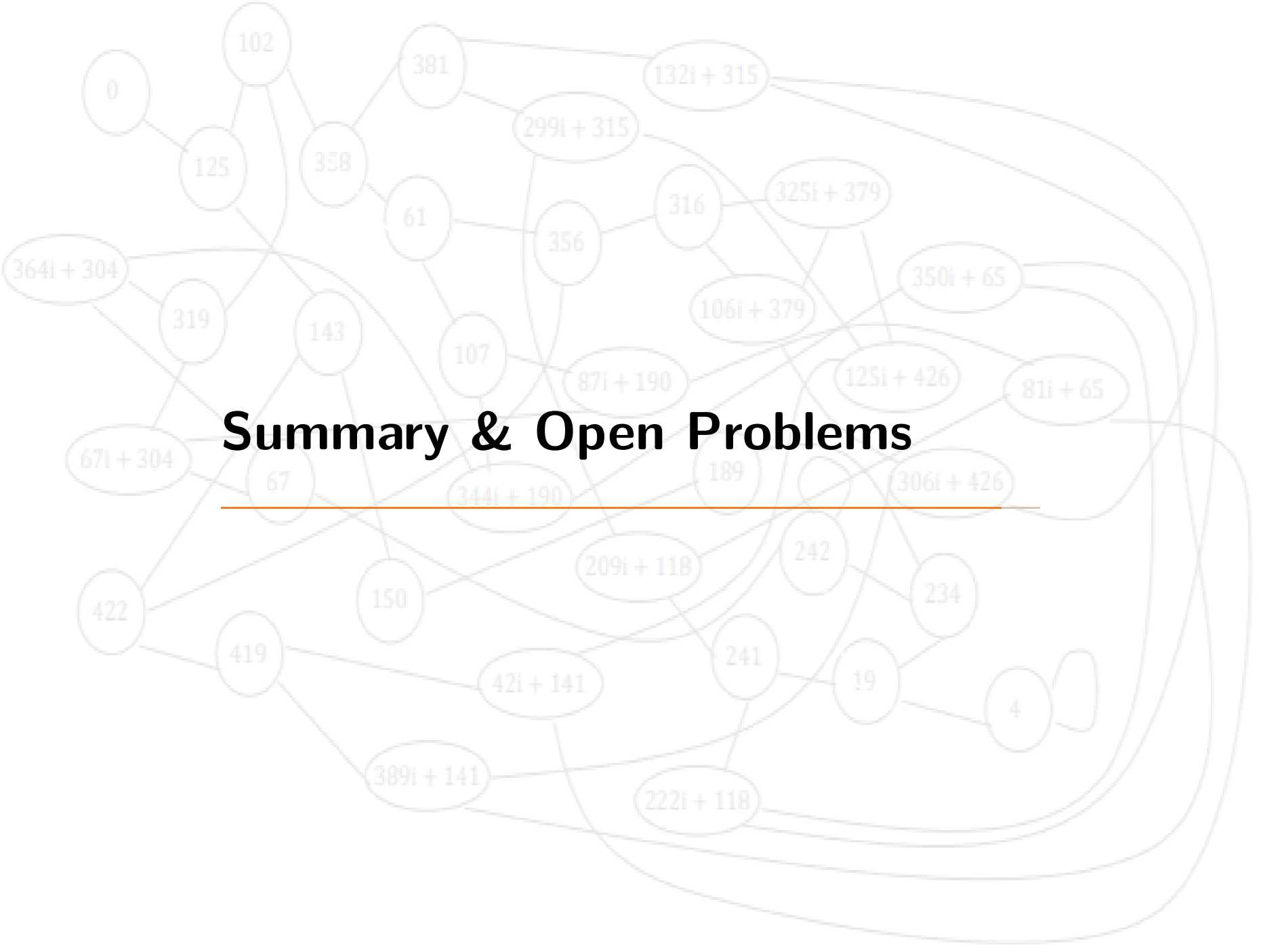
Given a supersingular elliptic curve E over $\mathbb{F}_{p^2}$ compute a non-trivial cyclic endomorphism of E of smooth degree.

The binding of our scheme follows as a reduction to this problem

Theorem

Our commitment scheme is computationally binding under the assumption that the Supersingular Smooth Endomorphism Problem for the curve E is hard.





Summary & Open Problems

Summary & Open Problems

Summary:

We show that isogenies can enter the world of commitment schemes and potentially be post-quantum

Delved into non-backtracking walks in isogeny graphs and introduced the mixing constant

Open Problems:

- Proving the conjectures on the upper bound of the mixing constant
- Understand non-backtracking walks through multiple isogeny graphs
- Applications of these mixing contexts in other settings
- Construct a homomorphic commitment scheme using isogenies

$$(g^m h^r)\left(g^{m'} h^{r'}\right) = g^{m+m'} h^{r+r'}$$

Thanks for listening
Questions?

ia.cr/2021/1031