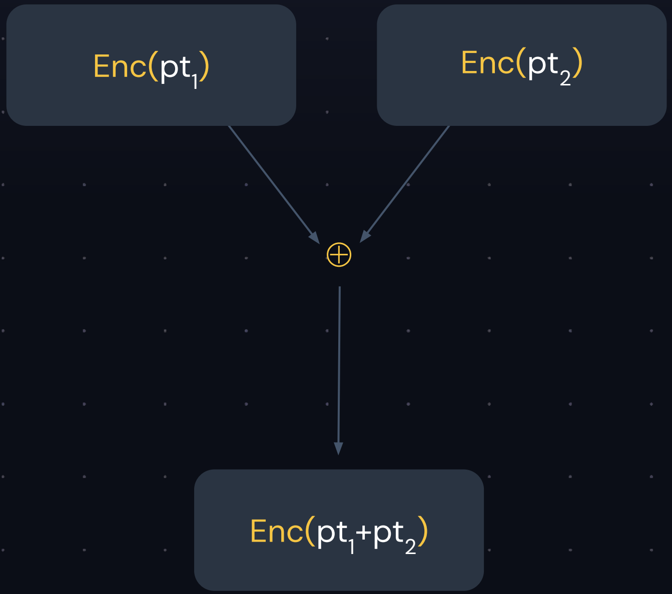
dWallet Labs

Tiresias: Large Scale, UC-Secure Threshold Paillier

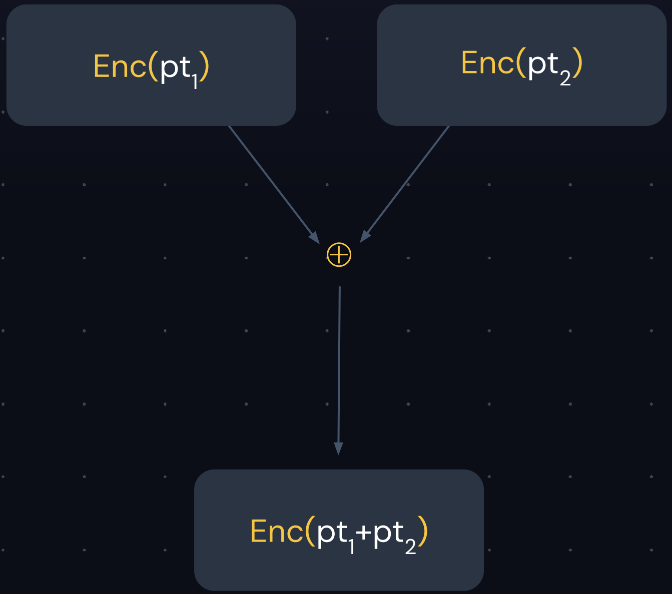
Quick Reminders

AHE: Additively Homomorphic Encryption

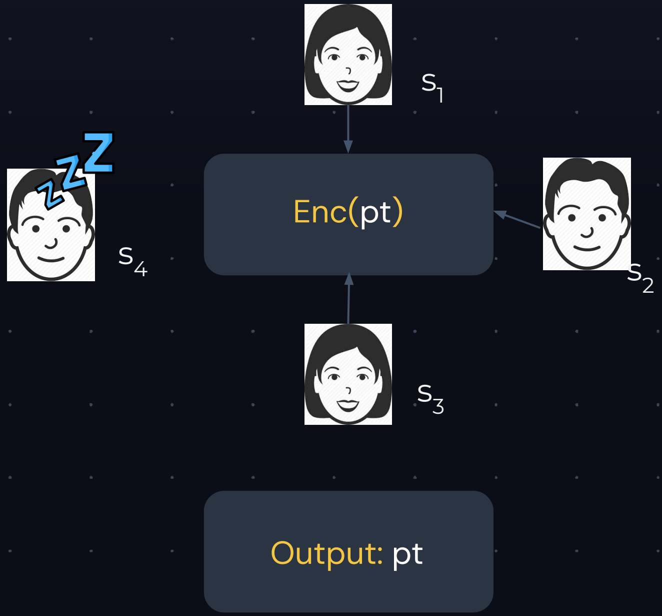


Quick Reminders

AHE: Additively Homomorphic Encryption

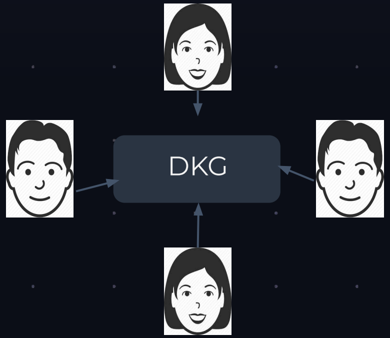


Threshold Encryption:



TAHE – Threshold Additively Homomorphic Encryption

DKG – Distributed Key Generation

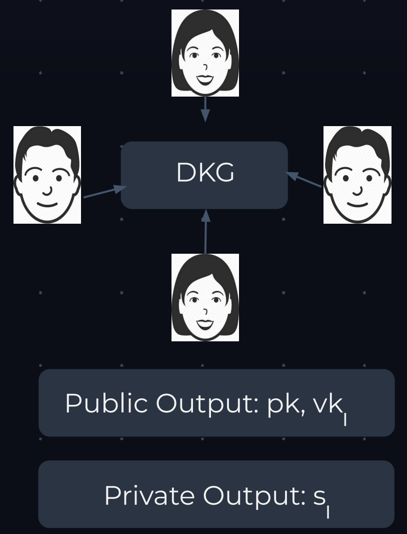


Public Output: pk, vk_i

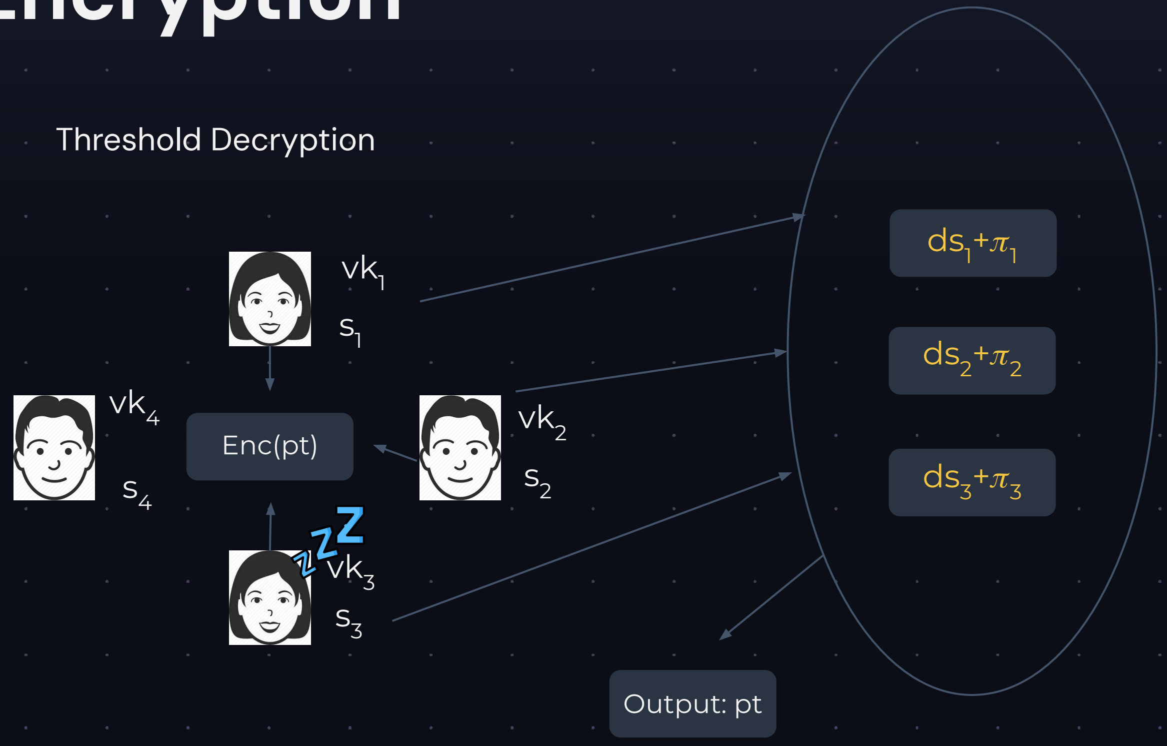
Private Output: s_i

TAHE – Threshold Additively Homomorphic Encryption

DKG – Distributed Key Generation



Threshold Decryption



What is it good for?

1. Voting Systems [FPS01, DJN10, KLM20]
2. Threshold Signatures Protocols [GGN16, FMM24+]
3. General Purpose MPC [BDTZ16]
4. Secret Maintenance On Blockchains

What is it good for?

1. Voting Systems [FPS01, DJN10, KLM20]
2. Threshold Signatures Protocols [GGN16, FMM24+]
3. General Purpose MPC [BDTZ16]
4. **Secret Maintenance On Blockchains**

The Paillier Cryptosystem

pk: $N=pq$ sk: $d \equiv 1 \bmod N$, $d \equiv 0 \bmod \phi(N)$

$\mathrm{Enc}(m;r)=(1+N)^m r^N \bmod (N^2)$

$\mathrm{Dec}(ct)=[ct^{sk} \bmod (N^2)-1]/N$

Threshold Paillier Encryption Assuming a Trusted Dealer

Trusted Dealer

pk: N=pq.
p and q are safe primes

s_i: Shamir Secret Sharing over Nϕ(N)

vk_i: v^{s_i}
v is a random quadratic residue

Threshold Paillier Encryption Assuming a Trusted Dealer

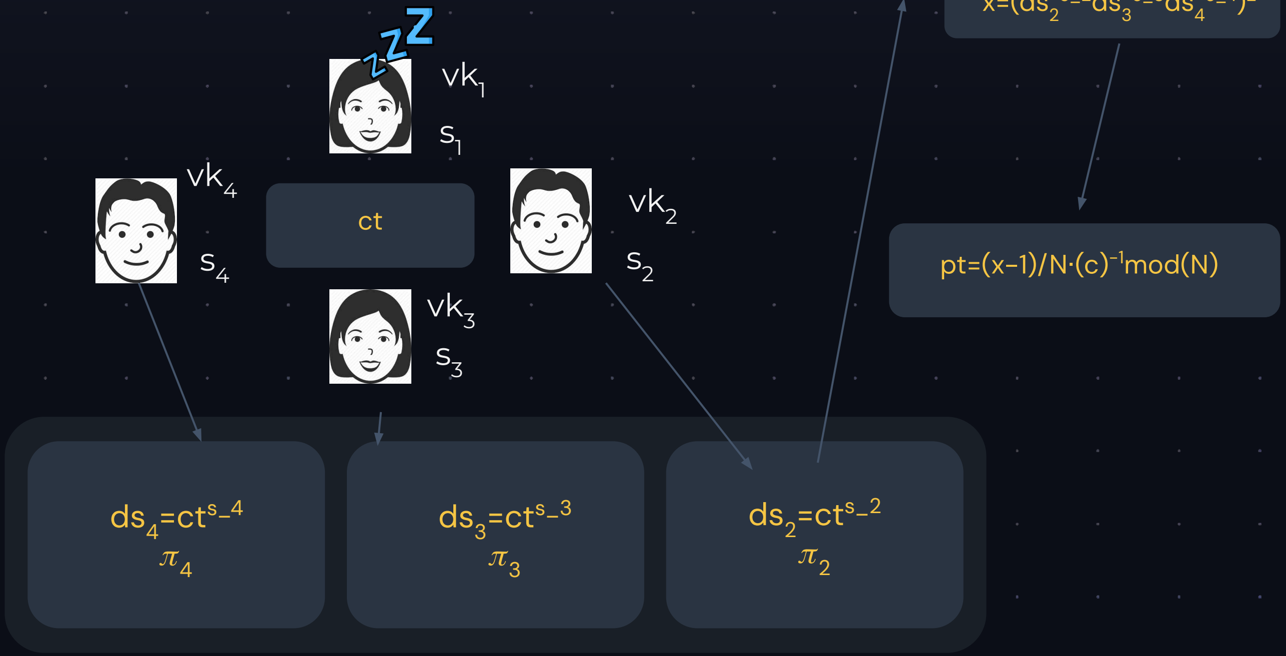
Trusted Dealer

pk: N=pq.
p and q are safe primes

s_i: Shamir Secret Sharing over $N\phi(N)$

vk_i: v^{s_i}
v is a random quadratic residue

Threshold Decryption



So what is the problem

We don't know how to practically
generate N which consists of Safe Primes.

Can you have a scheme with
practical DKG and efficient proofs?

What was Done up now?

	Key Generation	Proof Efficiency	Assumptions
[ACS02] Safe Primes	❌	✅	✅
[DK01]-Ad Hoc Assumptions	✅	❓ No Batching :(	❌
[FS01] B-Rough	❌	❌	✅
[HMR19] Cut-and-Choose	✅	❌	✅
This Work	✅	✅	✅

*[BDTZ16]: r-recovery, 2 rounds decryption

So what's So great about Safe Primes?

- QR_N is cyclic.

So what's So great about Safe Primes?

- QR_N is cyclic.
- Almost every element is a generator.

So what's So great about Safe Primes?

- QR_N is cyclic.
- Almost every element is a generator.
- For a small e if $x^e = 1 \bmod (N^2) \rightarrow x = 1 \bmod (N^2)$.

So what's So great about Safe Primes?

- QR_N is cyclic.
- Almost every element is a generator.
- For a small e if $x^e = 1 \bmod (N^2) \rightarrow x = 1 \bmod (N^2)$.
- Denote $\log_v(ct) = \alpha$. Specifically in the soundness proof we get $x = ds/vk^\alpha$ if $x = 1$ the statement is correct.

So what's So great about Safe Primes?

Nothing...

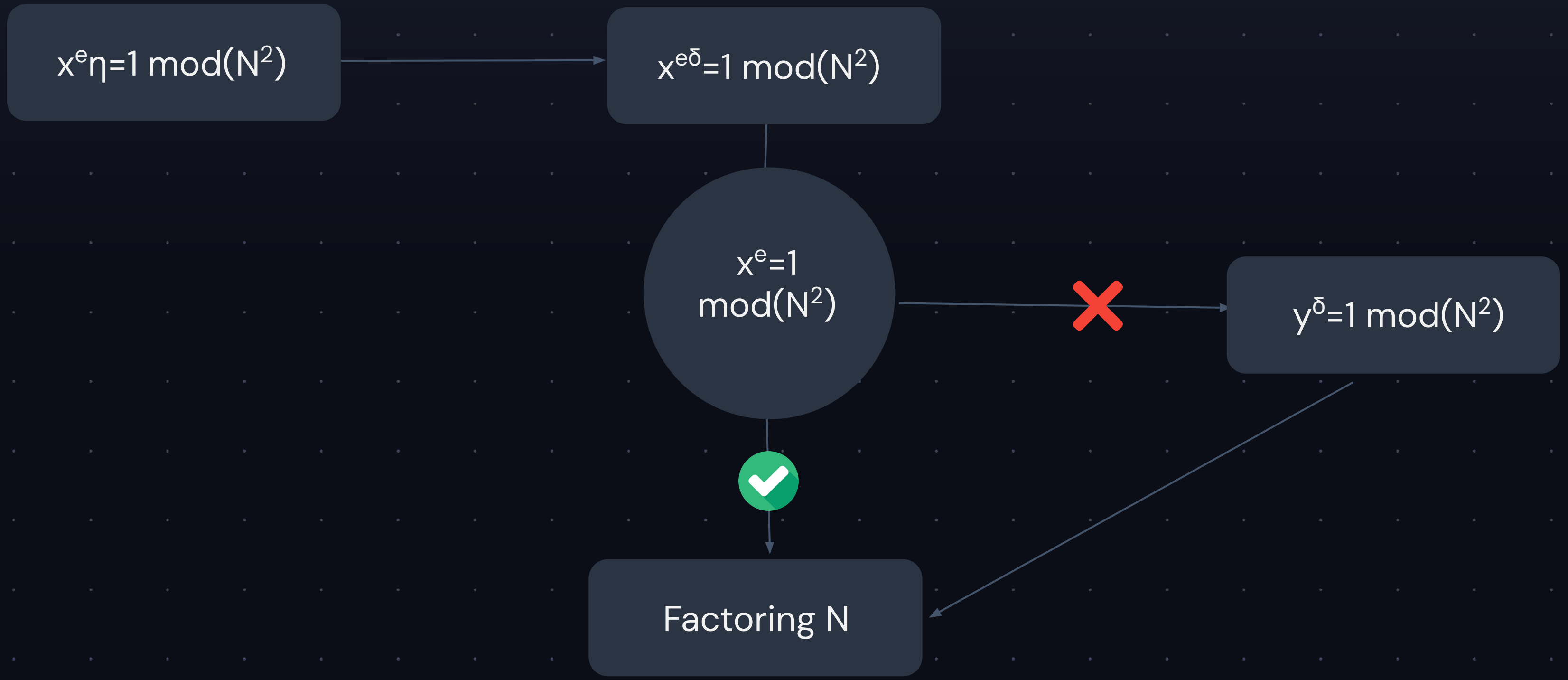
- **QR_N is cyclic.**
  $GCD(P-1,Q-1)=2$ is enough for this.

- **Almost every element is a generator.**
  There are Enough "Almost Generators".

- **For a small e if $x^e=1 \bmod (N^2) \rightarrow x=1 \bmod (N^2)$.**
  $x^e\eta=1 \bmod(N^2)$ gives either $x=1 \bmod (N^2)$ or allows for factorization of N

- **The Main Point:** There exists bad statement an adversary may be able to prove but finding them reduces to factoring.

Factoring N in case $x \neq 1$



Factoring N

Case – 1

1. Factor e
2. Remove powers of 2 from e (terminates in an odd number or a square root and thus factoring).
3. Exponentiate to the odd factors of e until you get 1. Denote the last non one value as y.
4. Calculating GCD(y-1,N) will give a non-trivial factor.

Case – 2

1. Factor using Pollard’s P-1 method

Alert! Alert! Non-Polynomial Reduction!!!
This is not a drill! I repeat this is not a drill!

Alert! Alert! Non-Polynomial Reduction!!!
This is not a drill! I repeat this is not a drill!

κ=128

σ=40

Alert! Alert! Non-Polynomial Reduction!!!
This is not a drill! I repeat this is not a drill!

$\kappa=128$

$\sigma=40$

$T(factor(e))\cong 42$

Alert! Alert! Non-Polynomial Reduction!!!
This is not a drill! I repeat this is not a drill!

$\kappa=128$

$T(factor(e))\cong 42$

$\sigma=40$

$P(g \text{ is } 2^{\sigma}\text{-almost generator})\cong 1-2^{\sigma}$

$T(\text{Pollard's } P\text{-}1)\cong 40$

Alert! Alert! Non-Polynomial Reduction!!!
This is not a drill! I repeat this is not a drill!

$\kappa=128$

$\sigma=40$

$T(\text{factor}(e)) \cong 42$

$P(g \text{ is } 2^{\sigma}\text{-almost generator}) \cong 1-2^{\sigma}$

$T(\text{Pollard's } P-1) \cong 40$

Practically this means factoring in very realistic times.

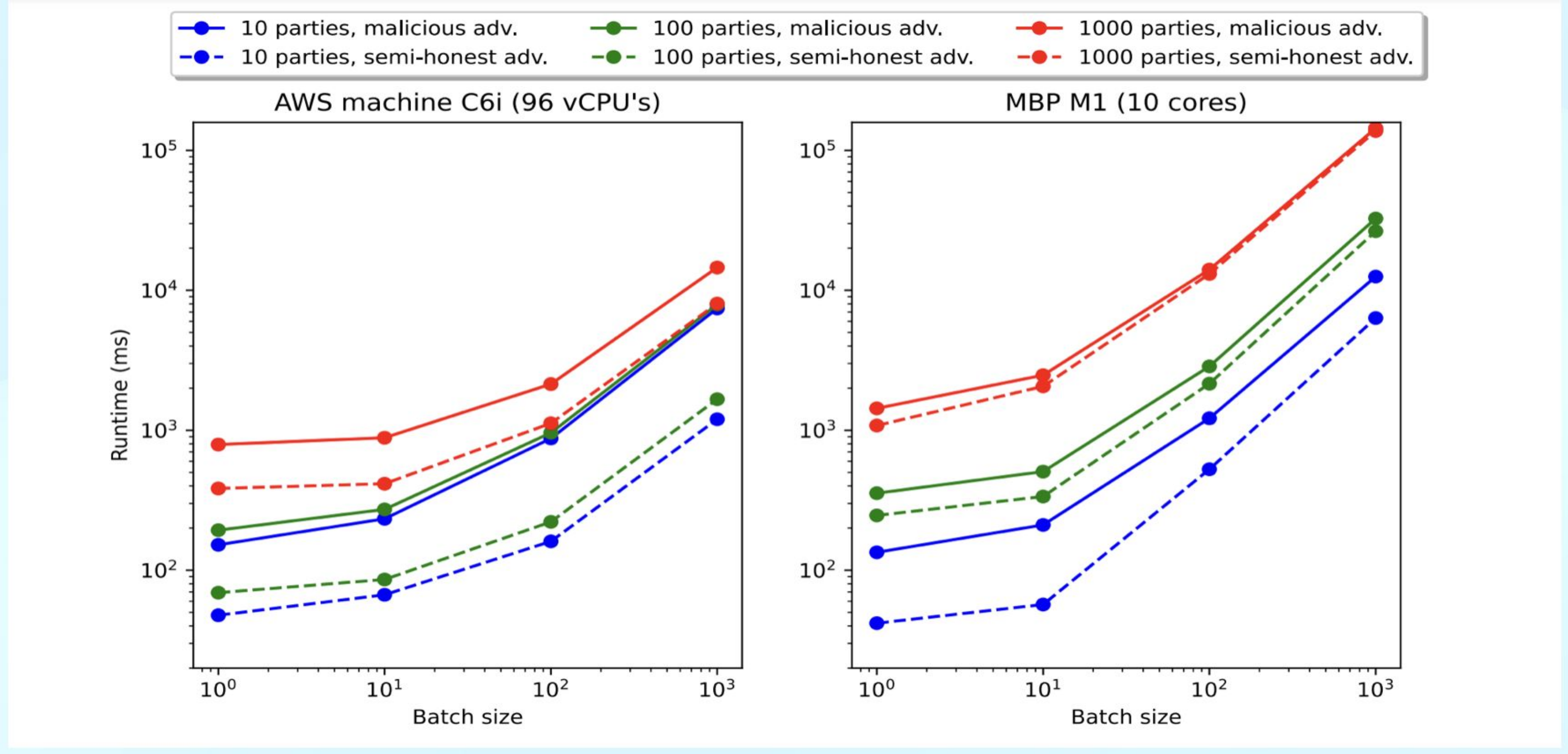
Moving to a Polynomial Reduction

- When forking send $e, e+1, e+2\ldots, e+2/\varepsilon$.
- Sample multiple bases and prove for each one to increase statistical security.

Supporting Batching

- Batching works via the small exponents method, i.e creating a random linear transformation from the proofs.
- We use similar reduction techniques to prove its security.
- We show "round-by-round soundness" to avoid security loss in the Fiat-Shamir transformation.
- Batch Verification works similarly.

Implementation



Thank you for listening!

Thank you for listening!

Questions?

Is Paillier Still Relevant when Class Groups Exists?

- Simplicity = Security*
- Implemented for cryptographic use
- Well established RSA adjacent assumption (DCR)
- Every group element is a valid ciphertext
- Efficient hash to group