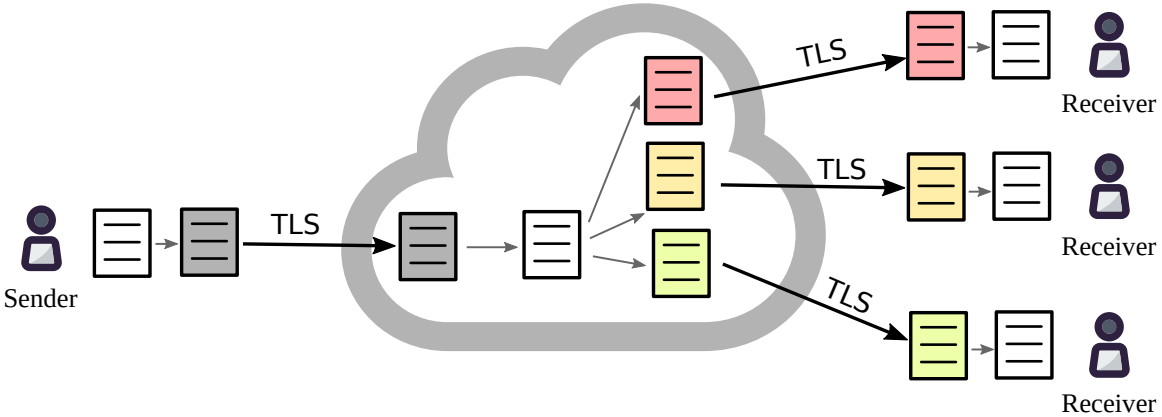
Decentralized internet: beyond public key encryption

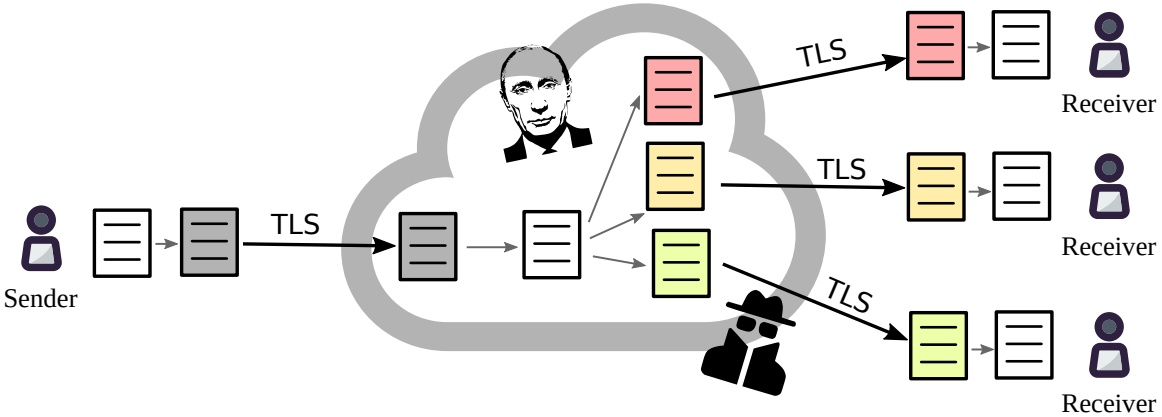
NuCypher

Michael Egorov, CTO

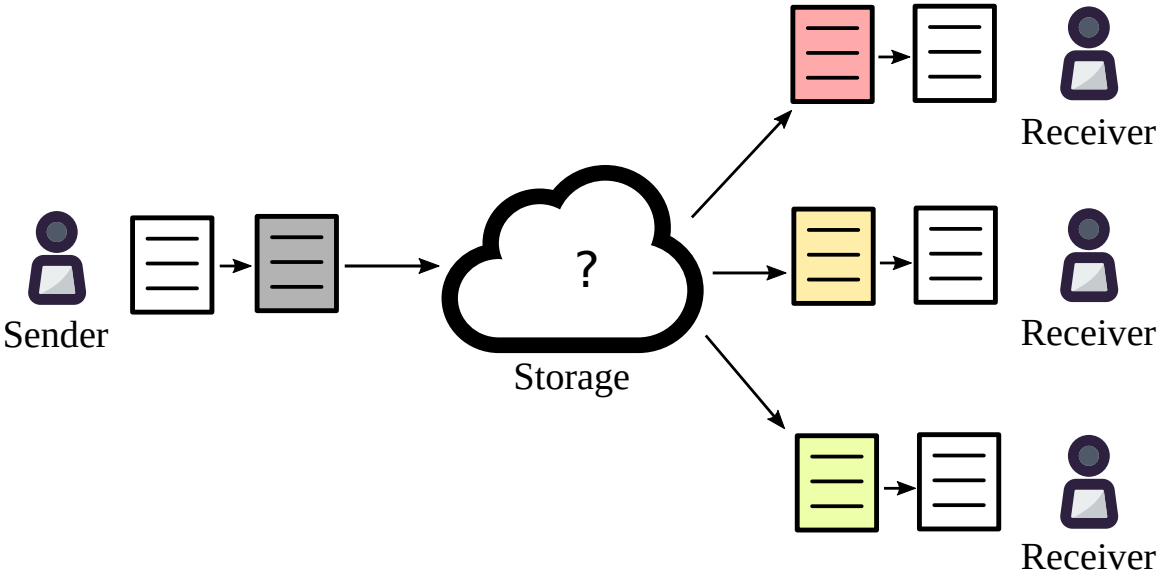
Transport encryption in traditional internet



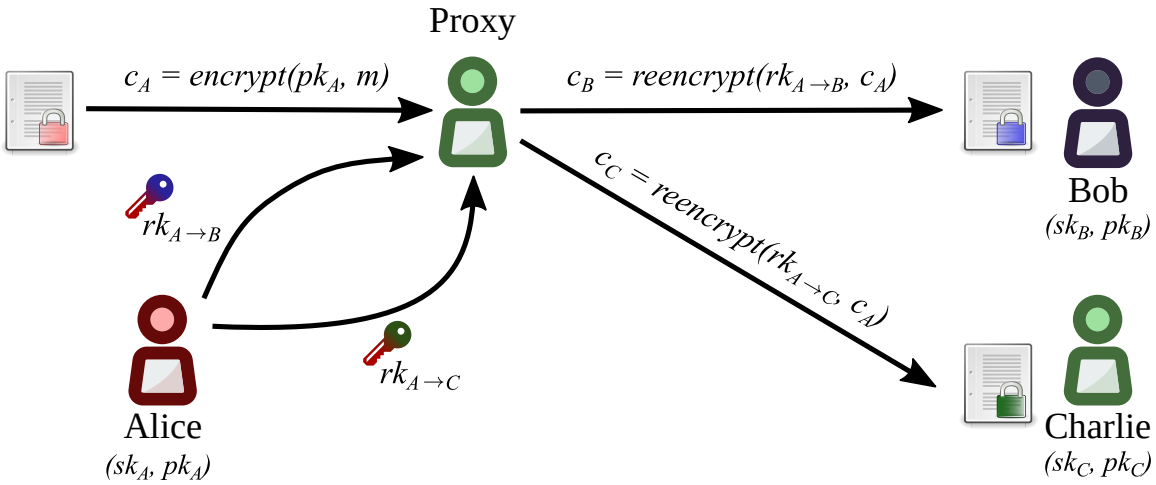
Transport encryption in traditional internet



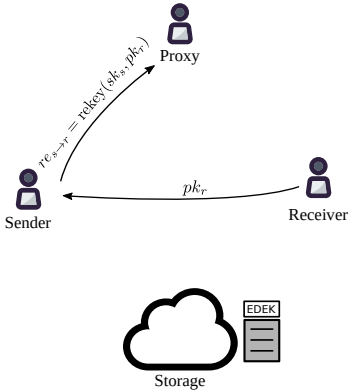
Does TLS work when decentralized?



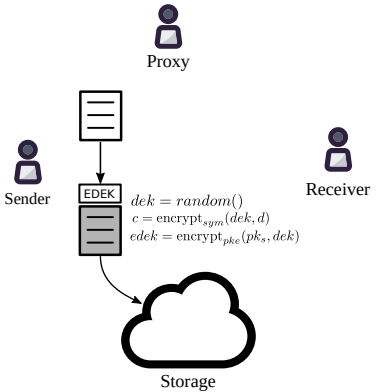
Proxy re-encryption (PRE)



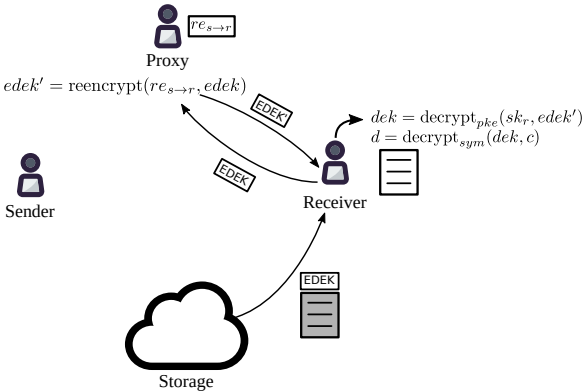
Decentralized permission management via PRE



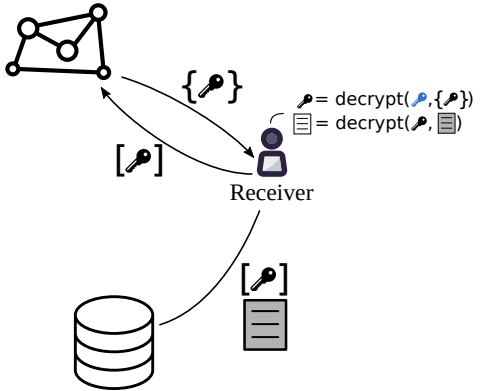
Decentralized permission management via PRE



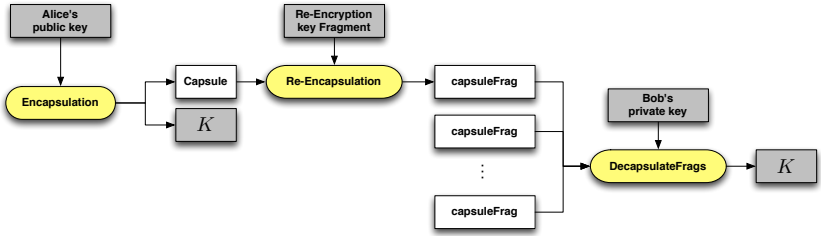
Decentralized permission management via PRE



Decentralized permission management via PRE



Umbral: Threshold Proxy Re-encryption



- Reference implementation: `https://github.com/nucypher/pyUmbral`
- Documentation: `https://github.com/nucypher/umbral-doc`

Umbral: Threshold Proxy Re-encryption

- “Umbral” is Spanish for “threshold”
- PRE properties: Unidirectional, single-hop, non-interactive
- Follows a KEM/DEM approach:
  - UmbralKEM provides the threshold re-encryption capability
  - Uses ECIES for key encapsulation with ZK proofs of correctness for verifiability on prime order curves (such as secp256k1)
  - DEM can be any authenticated encryption (currently ChaCha20-Poly1305)
- IND-PRE-CCA security
- Key splitting is analogous to Shamir Secret Sharing
- Verification of re-encryption correctness through Non-Interactive ZK Proofs
- Reference implementation: `https://github.com/nucypher/pyUmbral`
- Documentation: `https://github.com/nucypher/umbral-doc`

How to go beyond sharing

Multi-Party Computation

- Interactive protocol
- Slow Performance

Fully Homomorphic Encryption

- Slow Peformance
  - NuCypher has developed a GPU-accelerated FHE library: nuFHE

Non-interactive Zero-Knowledge proofs

- Actively researched now
- Useful even beyond privacy-preserving value transfers

Oblivious RAMs

- For hiding access patterns
- Didn't receive much of attention by DApps yet

Fully Homomorphic Encryption

nuFHE library

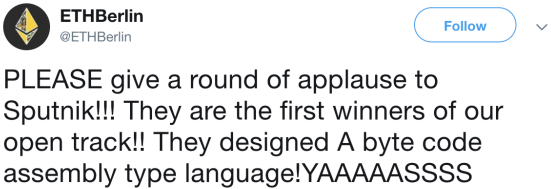
- Based on TFHE: Fast Fully Homomorphic Encryption over the Torus
- GitHub: `https://github.com/nucypher/nufhe`
- GPU implementation of fully homomorphic encryption
- Uses either FFT or integer NTT
- Achieved 100x performance over TFHE benchmarks

Platform	Library	Performance (ms/bit)	
		Binary Gate	MUX Gate
Single Core/Single GPU - FFT	TFHE (CPU)	13	26
	nuFHE	0.13	0.22
	Speedup	**100.9**	**117.7**
Single Core/Single GPU - NTT	cuFHE	0.35	N/A
	nuFHE	0.35	0.67
	Speedup	**1.0**	**-**

FHE Proof of Concept

Sputnik

- GitHub: `https://github.com/nucypher/sputnik`
- Assembly language and interpreter for FHE that uses nuFHE
- Commits a merkle root of computation to the blockchain for proof of logic flow
- Used to execute first homomorphic smart contract at ETHBerlin 2018



Can DApps propel privacy-preserving computations research and adoption?

- CPU: 70 ops/s;
- GPGPU: 7000 ops/s;
- FPGA: ???;
- ASIC: 1M ops/s?;
- Optical computers: on par with CPUs while keeping data encrypted?
  - Hint: a thin lens can do FFT

Conclusion and references

NuCypher

Website: https://www.nucypher.com
Whitepaper: https://www.nucypher.com/whitepapers/english.pdf
Proxy Re-encryption Network: https://github.com/nucypher/nucypher
Umbral Reference Implementation: https://github.com/nucypher/pyUmbral
nuFHE: https://github.com/nucypher/nufhe
Discord: https://discord.gg/7rmXa3S
E-mail: michael@nucypher.com
E-mail: hello@nucypher.com