UNBOUND

[WHERE SECURITY IS KEY]

Settings and Considerations for Standardizing Multi-Party Threshold Schemes

Yehuda Lindell
Unbound Tech and BIU

This Talk

- This will be a talk that primarily poses questions
- Part 1 – diverse applications for threshold cryptography already today
- Part 2 – questions that arise in different scenarios

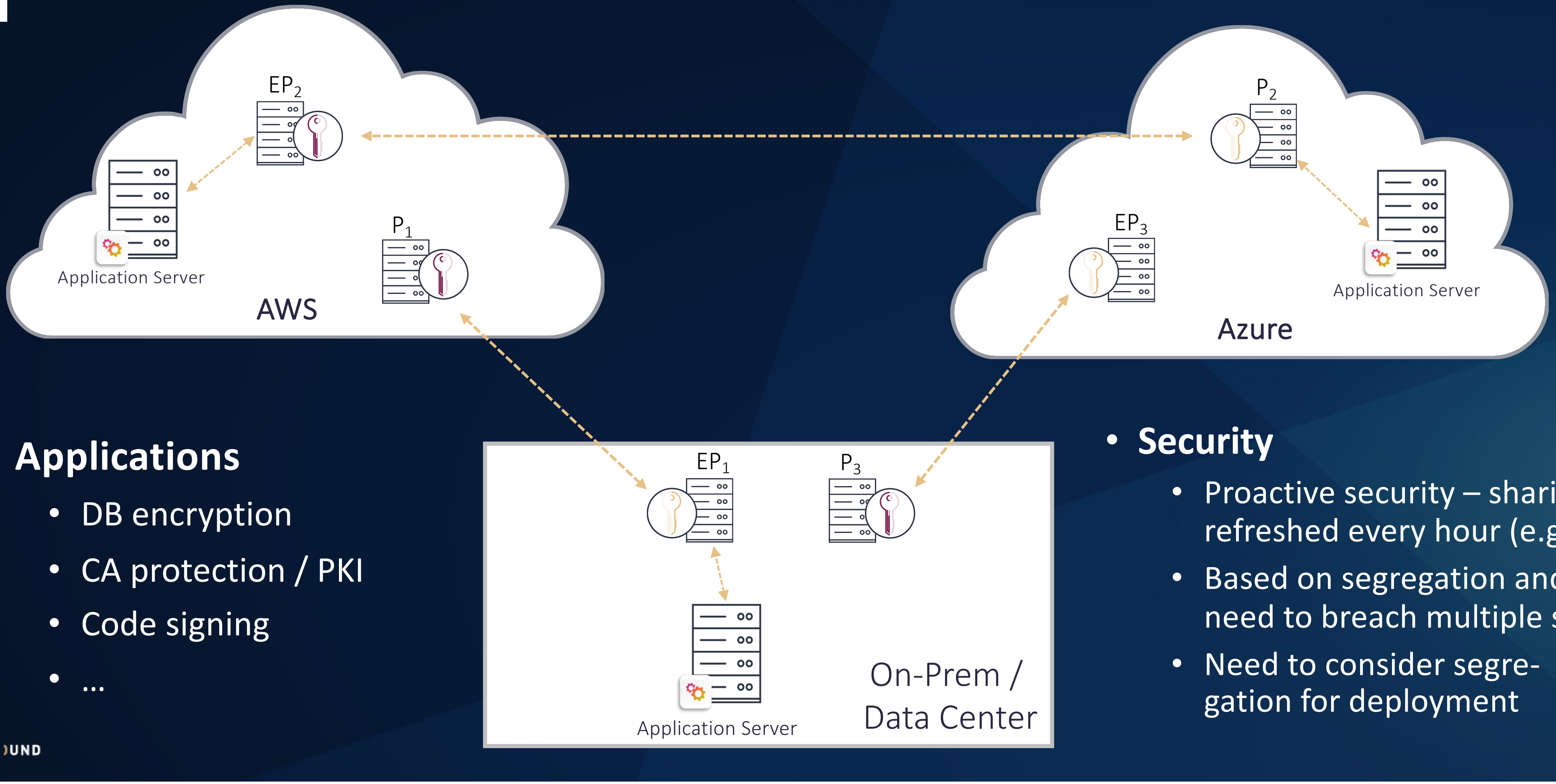
Applications Today

- **A lot of interest in threshold crypto today is coming from the cryptocurrency space**
  - There is very strong product-market fit for this use case
- **Even here there are distinct use cases that require different setups**
  - Custody vs exchange, full control in the organization vs split, end-user wallets
- **I want to talk about other applications that are in use today**

HSM “Replacement”

- **Hardware Security Modules (HSMs) protect keys by using them inside and never revealing them**
  - Strong physical protection against tampering, physical side-channels, etc.
  - Cannot run other code alongside, so isolation against software side channels
- **HSMs are a pain**
  - They all work differently
  - They are a physical anchor in a virtualized world
  - They often require physical presence for administration (PED)
- **HSM security isn’t as clear as one may think**
  - Primarily security against hardware attacks
  - Major vulnerabilities have been shown

Deploying an MPC-Based Virtual HSM



- **Applications**
  - DB encryption
  - CA protection / PKI
  - Code signing
  - …

- **Security**
  - Proactive security – sharing is refreshed every hour (e.g.)
  - Based on segregation and need to breach multiple sites
  - Need to consider segregation for deployment

Key Theft vs Key Misuse

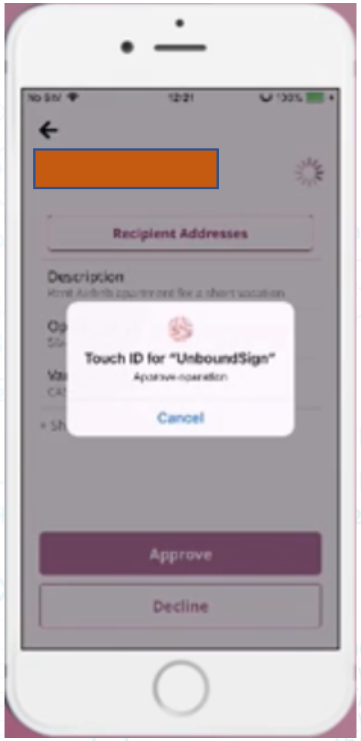
- **Legacy solutions focus on preventing key theft**
  - Cryptographic key is never exported from device
- **But, anyone accessing the machine who is authorized to carry out operations can also carry out operations**
  - Makes sense for application-layer credit card encryption
  - Very problematic for
    - Code signing
    - Transaction signing
    - More
- **MPC provides solutions for preventing key misuse**

Key Misuse Protection via MPC

- **Consider quorums of signers**
  - Policy checkers – time of day, rate limiting
  - Identity – authorized for operation
  - Anti-fraud / risk check
  - Human approvers (where relevant)
- **A similar thing can be used for encryption:** policy checkers, anomaly detection, etc.
- **Maker-checker workflows**

Authentication with MPC

- **Virtual smartcard, OTP token on mobile**
  - Mobile and server hold key shares and compute via MPC
  - Refresh key sharing at *every single operation*
    - Proactive security
  - All operations are audited at the server as well as mobile
    - Full visibility into operations
  - Mobile is always with the user (usability and security)
- **Similar effect for endpoints (laptops/servers)**



[redacted]

Recipient Addresses

Description
Rent Airbnb apartment for a short vacation

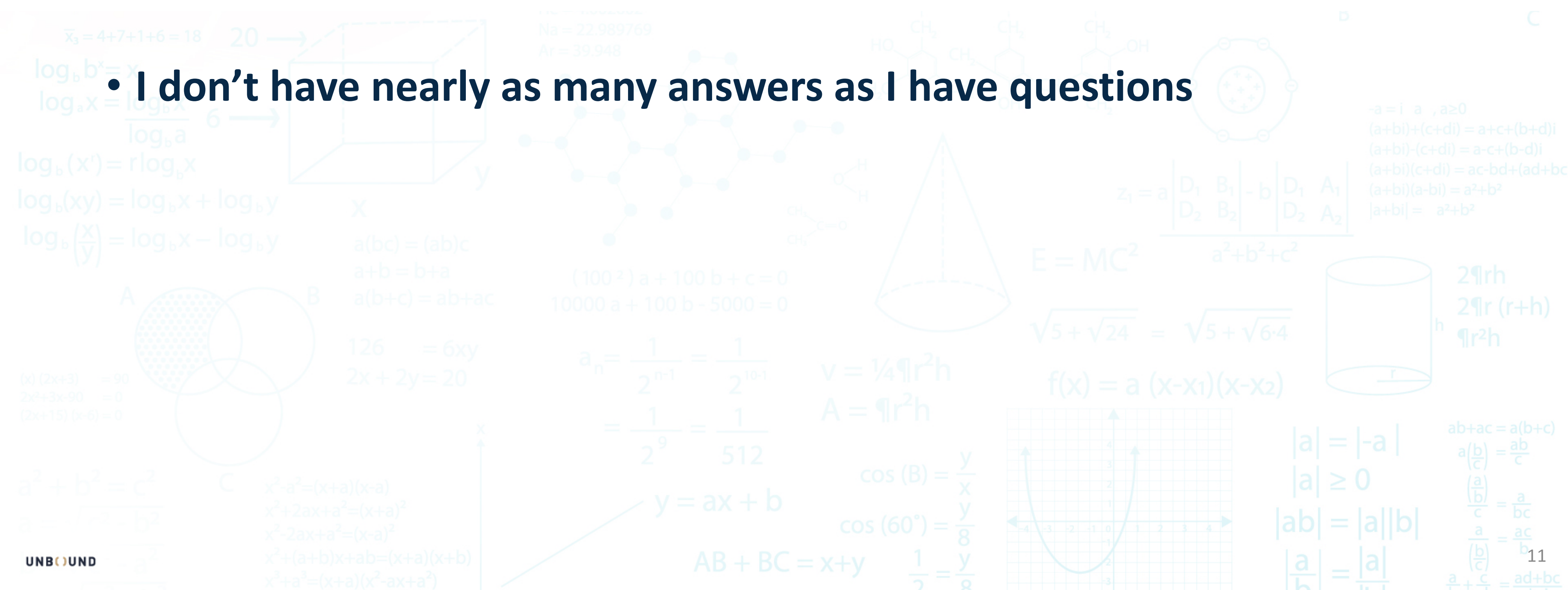
Touch ID for "UnboundSign"
Approve operation

Diverse Settings

- **Key belongs to different entities or same entity**
  - How do different entities collaborate (need same or compatible software)?
- **Entity can be server (always connected), mobile, belonging to human or organization, and more**
- **Threats can be different (cloning, key theft, key misuse,…)**
- **Different settings require different properties:**
  - Installation and setup
  - Backup
  - Threat analysis

Questions

- **There are many questions that arise in different scenarios**



- **I don't have nearly as many answers as I have questions**

Standardization Levels

- **Three major levels**
  - Standardization of basic primitives
    - Garbled circuits, OT, secret sharing, etc.
  - Standardization of full protocols
    - 2-party AES, multiparty ECDSA, RSA key generation, etc.
  - Standardization of definitions and methodology
    - Malicious, proactive, full proofs of security

Standardization Level Questions

Basic Primitives

- Gap between basic primitive and full solution is huge
- Without high expertise, very hard to build a secure MPC protocol, even from secure primitives
  - Bigger gap than for AES
- Necessary but not sufficient

Full Protocols

- Still very dynamic – standardize RSA key generation and next year 10-fold improvement
- Many scenarios and different protocols needed for all
- Could choose most popular and at least achieve 80%

Definitions & Methodology

- Who validates the proof?
  - Is this viable at all?
- Different settings require different levels of security
  - Should we require malicious always?
- Standardization bodies don't work in this way
  - Would be like saying – "any encryption is fine as long as it's CCA secure"

Security Architecture

- **We talk a lot about standardizing the MPC core, but this is far from what makes the system secure**
- **Standardization of security architecture is very important**
  - How is the system set up and bootstrapped?
  - How are shares shared?
    - If less than a quorum can add parties, then easy to bypass the quorum
  - How is the system backed up safely?
  - How can additional pairs/sets of machines be added
    - Challenge of preventing a single point of failure
  - Do we need to standardize segregation elements?
    - Different OSs, different admins, different environments? Very impractical!
- **FIPS includes elements of security architecture today**

Assumptions

- **Ideally, X should rely only on X**
  - Threshold ECDSA should rely only on the security of ECDSA
- **More practically, X should rely only on NIST certified primitives**
  - Threshold ECDSA should rely only on ECDSA, AES and SHA256
  - Can it rely on DDH? This is implicit in NIST certified primitives
- **What about other assumptions?**
  - Paillier, lattices, Bilinear maps
  - Are these the same as each other? Who determines?
  - Can we use new assumptions about class groups in a product?
- **What about multilinear maps and obfuscation assumptions?**
  - When is an assumption ready for use in production?
- **What about sub-exponential hardness?**

Models

- **What models are acceptable?**
  - Standard model
  - Random-oracle model
  - Generic group model
  - Knowledge of exponent assumptions
  - Sub-exponential or quasi-polynomial simulation

- **Is everything acceptable? Are there preferences? Can standardization deal with preferences?**

Adversarial Power

- Should standardization mandate malicious, covert or semi-honest?
- Should standardization mandate pro-active security?
- I am a strong advocate of malicious pro-active, but should this be mandated?
- What if a user utilizes a trusted execution environment that it accepts as reliable? Can it then run semi-honest?

Security Model & Composition

- **Should standardization mandate the security model?**
  - Game-based vs simulation
  - Stand-alone vs concurrent composition
  - Concurrent self-composition vs general composition / universal composability
- **What assumptions are reasonable for composition?**
  - Fiat-Shamir is very popular, but it actually requires rewinding
  - Can we rely on knowledge of exponent assumptions to remove rewinding?
    - Requires a more expensive protocol
  - Can we just assume that a Sigma-protocol with FS is NIZK or NIZKPoK?
    - The rewinding is needed to ascertain that it's a NIZK, so can we just then assume it?

Summary

- **These questions and more actually come up**
  - In our internal discussions and design
  - Sometimes when we do independent cryptographic review
  - Sometimes when customers bring cryptographers to do independent review
- **Standardization should take the most flexible, least common denominator, that is considered “secure”**
  - This is extremely complex for threshold crypto today
  - The fact that there are diverse settings means that there are different needs, and they make a big difference
- **I want to reiterate from yesterday that other FIPS standardization efforts should take threshold crypto compatibility into account**
  - Irrespective of standardizing threshold crypto itself

UNBOUND

[WHERE SECURITY IS KEY]