Practice Based Recommendations for Standardization of Threshold Cryptography

Dan Shumow MSR Security & Cryptography

NIST Threshold Cryptography Workshop 2019

March 12th, 2019

Outline

- Introduction
- Examples of Scenarios for Threshold Secret Sharing
- Conclusions
- Recommendations

Introduction

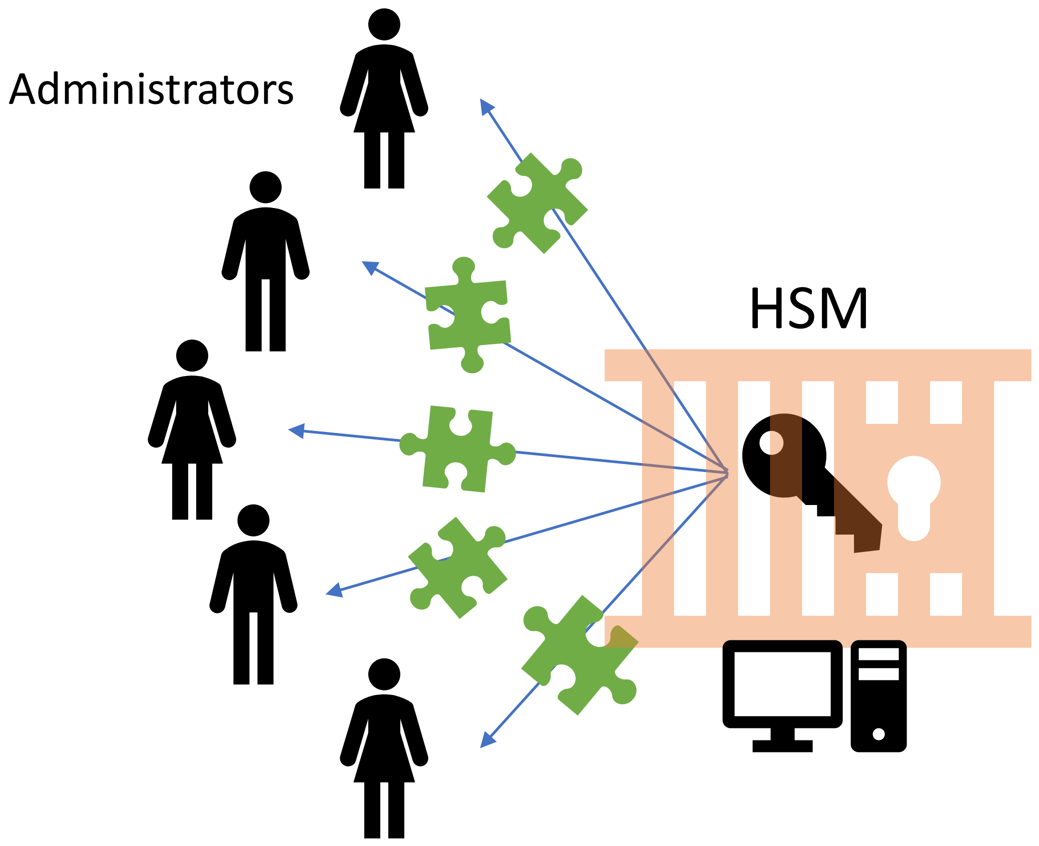
- MSR Security & Cryptography team provides consulting to Microsoft products and services on cryptographic solutions.
- We have consulted with several product teams about using threshold secret sharing.
  - No teams have ended up using this as a solution.
- Generally speaking there is quite a bit of interest in threshold cryptography.
  - Hopefully our examples provide some perspective on what, specifically, is needed.

Examples

- HSM Key Backup
- Administrator Credential Recovery
- Bitcoin wallet
- Disaster recovery of hard drive backup
- Transportation of high value VHD

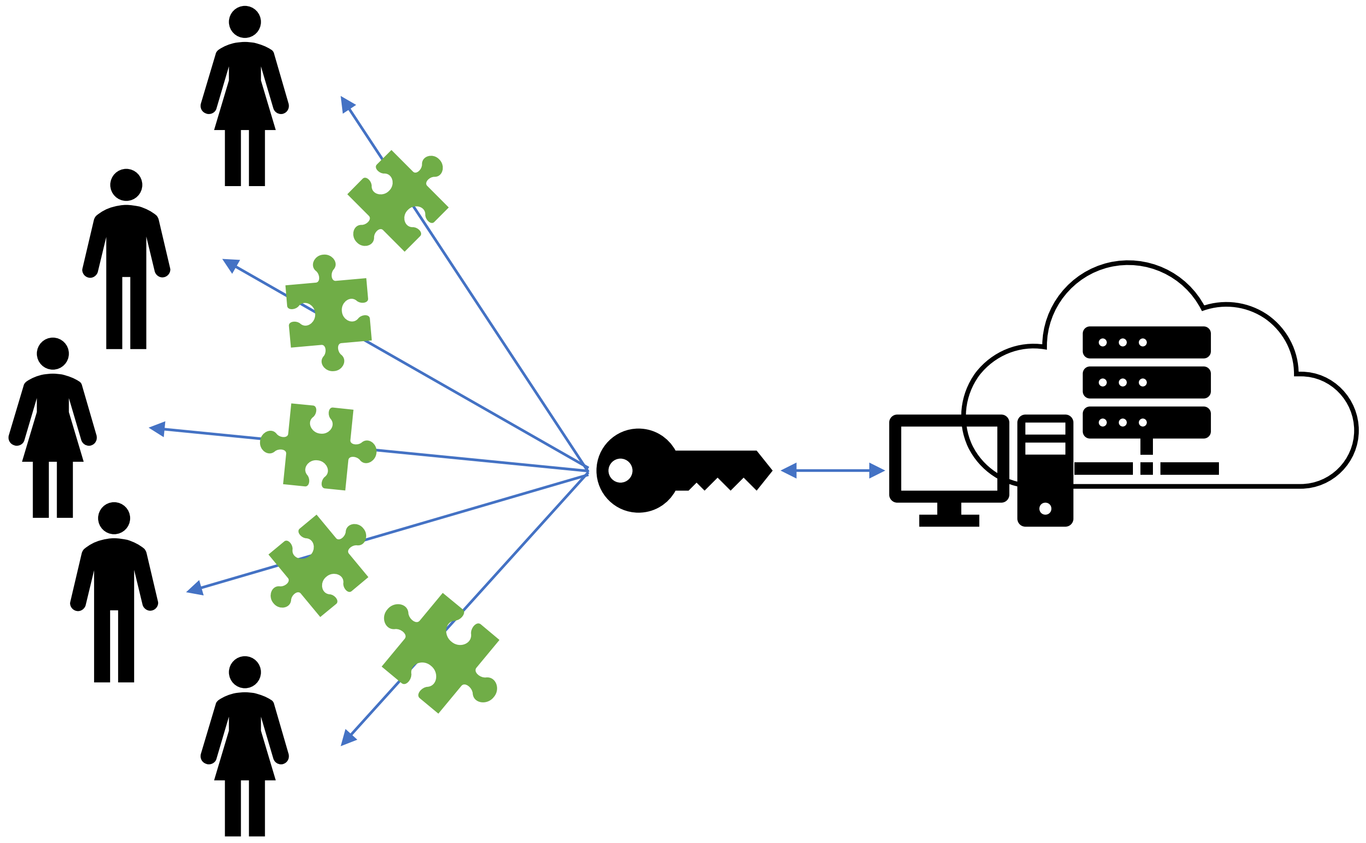
Example: Robust HSM Key Backup

- To robustly and securely backup a key in an HSM this team wanted to use threshold secret sharing.
- We proposed using Shamir threshold secret sharing.
- HSM in question protected exported keys with admin smartcards, any of which could be used to reload the key.
- **Shamir secret sharing did not remove that bottleneck and was thus deemed not necessary.**



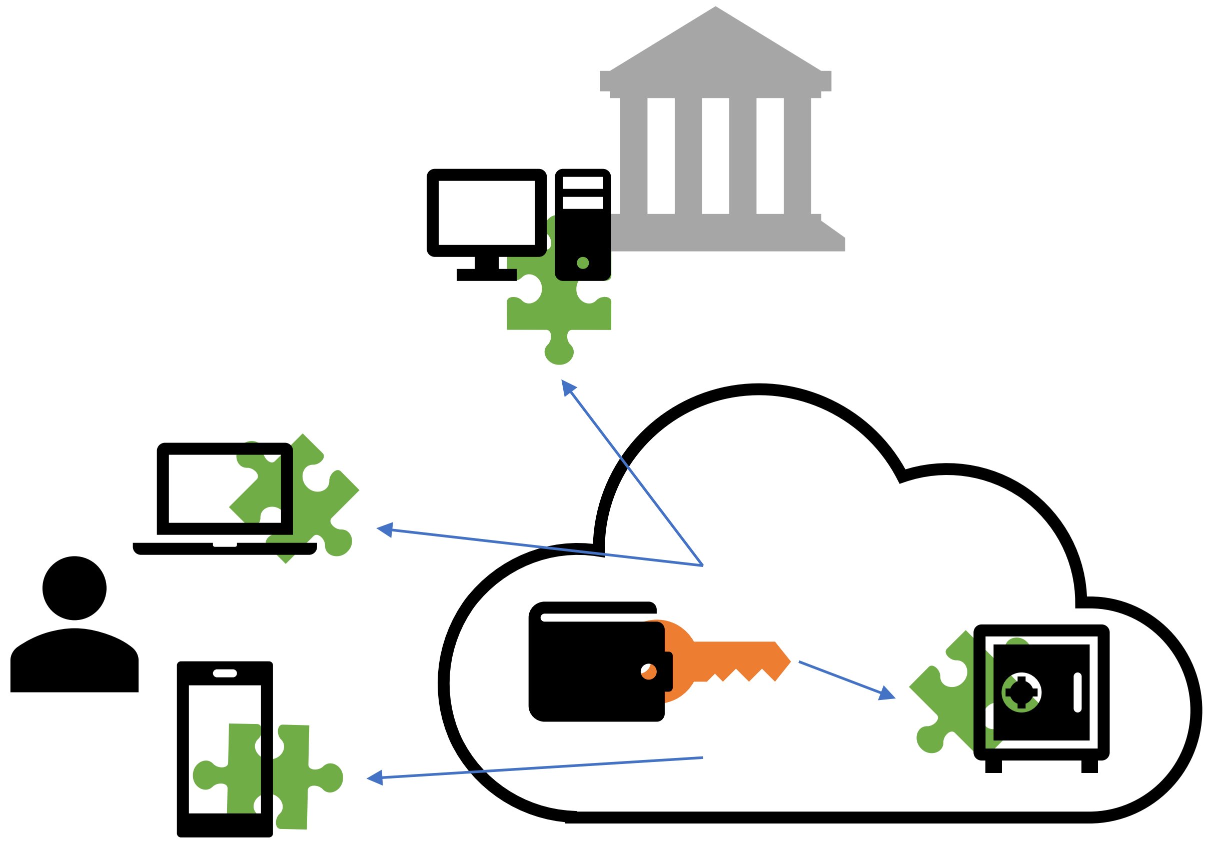
Example: Administrator Credential Recovery

- Administrator credentials (password) for a production server shared between a set of administrators, with threshold enforcing a quorum for recovery.
- Problems/objections:
  - No UI/utility to perform full protocol.
  - Team did not have a secure computer to execute code on.
  - Single point of failure.
- **Entire scheme abandoned.**



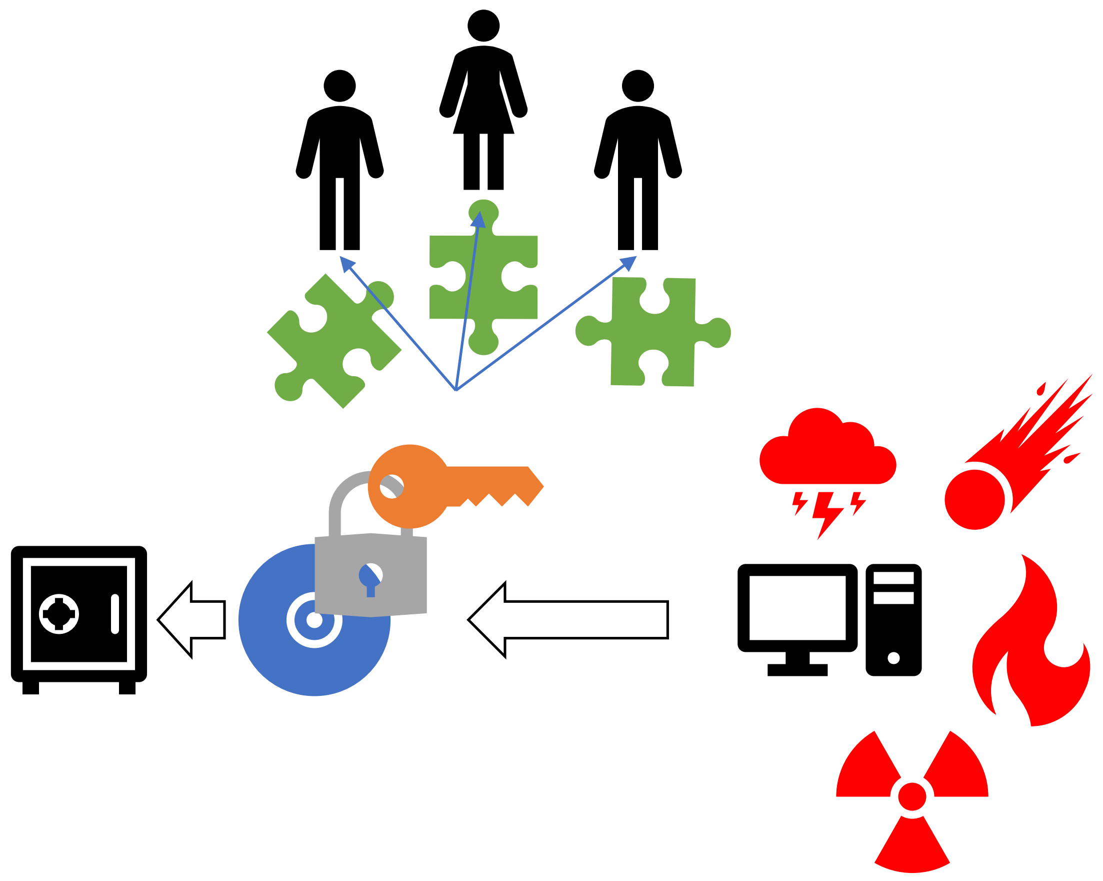
Example: Bitcoin Wallet

- Team designing a bitcoin wallet solution, investigated threshold secret sharing.
- This was prior to recent threshold ECDSA advances.
- Idea: Secret split between cloud, bank backup, with secret recombined in protected hardware (enclave or TPM 2.0.)
- **Project was canceled.**



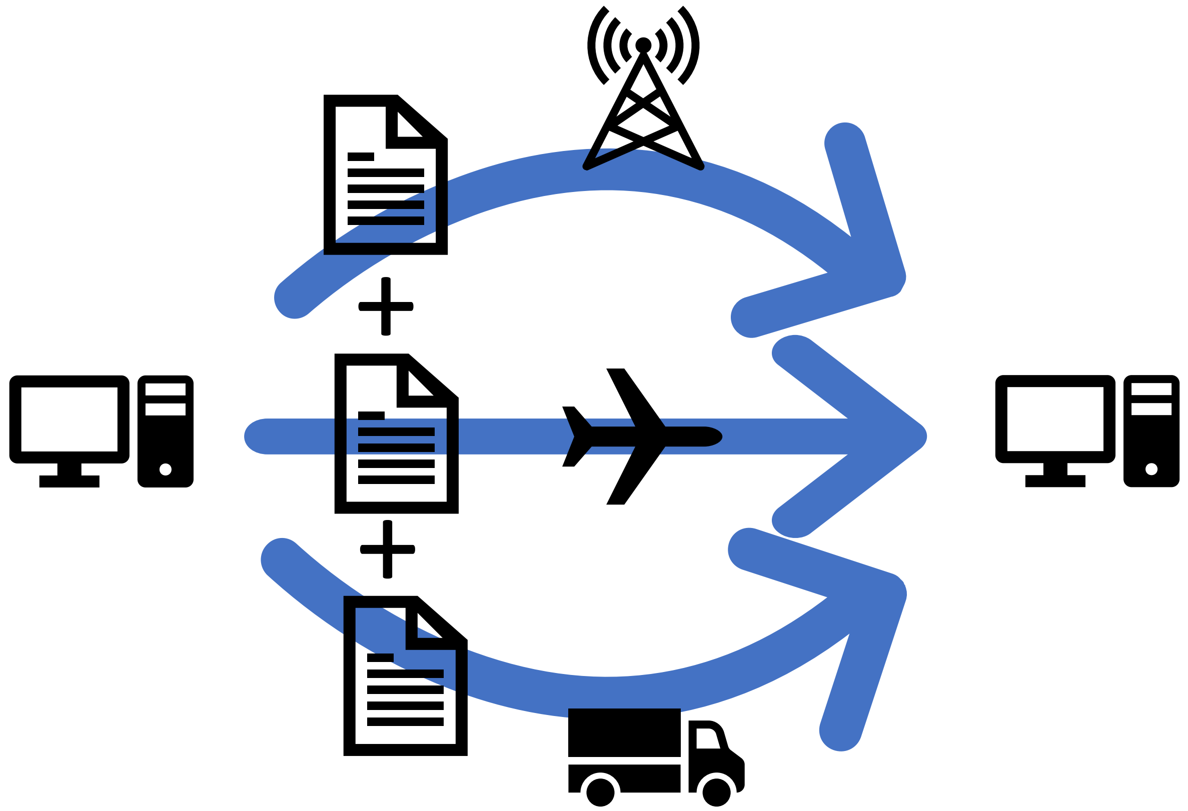
Example: Disaster Recovery Key Protection.

- Hard drive backup encryption key shared with threshold secret sharing, shared among administrators/operators.
- Objections:
  - Not standardized.
  - No UI/Utility to perform full protocol.
  - Team didn't want to make decisions about data formats for storing secrets.
- Team decided benefit of solution not worth it.



Example: Transportation of High Value VHD

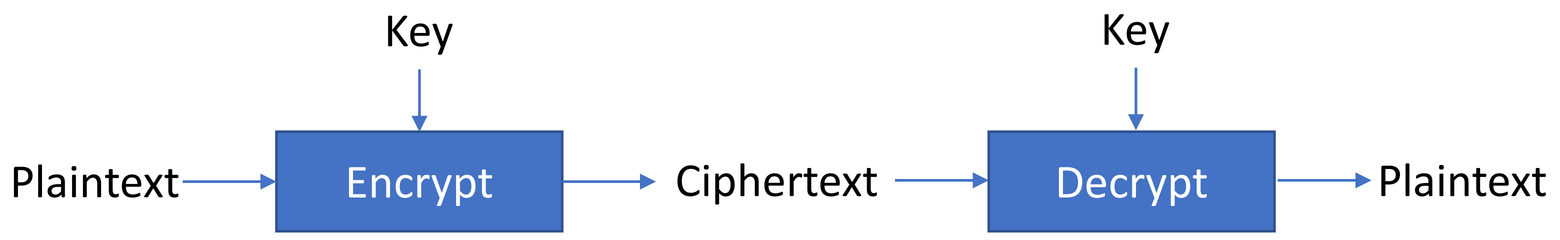
- An operations team tasked with transporting a “high value” VHD to another location.
- Wanted $t = n$, i.e. one file lost, all files would be deleted.
- Simple secret splitting sufficed.
- **This was the only solution we proposed that was actually used by a team!**

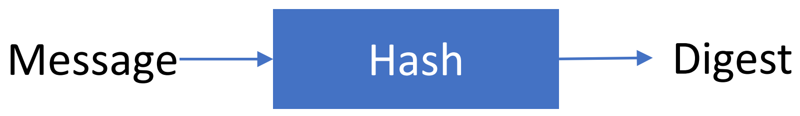


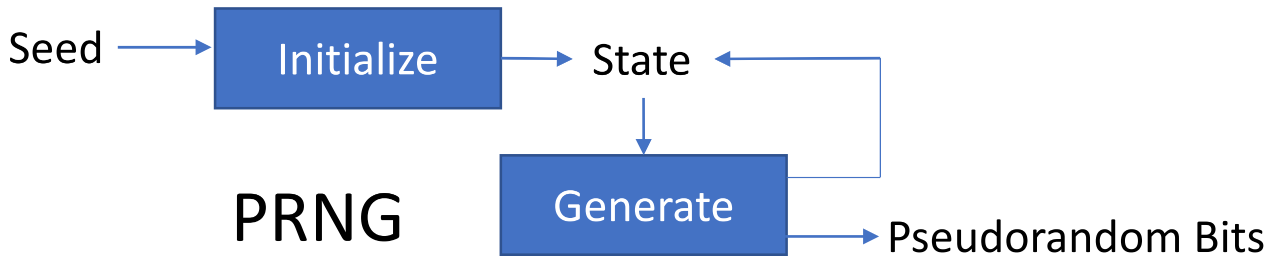
Threshold Secret Sharing: Not an Easy Sell

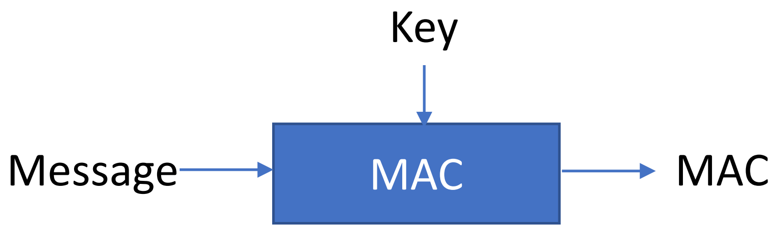
- Implementing, deploying and maintaining software components to perform this was viewed as too expensive.
  - General purpose software engineers don’t want to do fancy crypto.
  - Wanted an end-to-end solution provided or maintained by another team.
- Lacked clear guidance on security necessities for environment for recombining secret.
- Did not seamlessly integrate with previous hardware solutions.

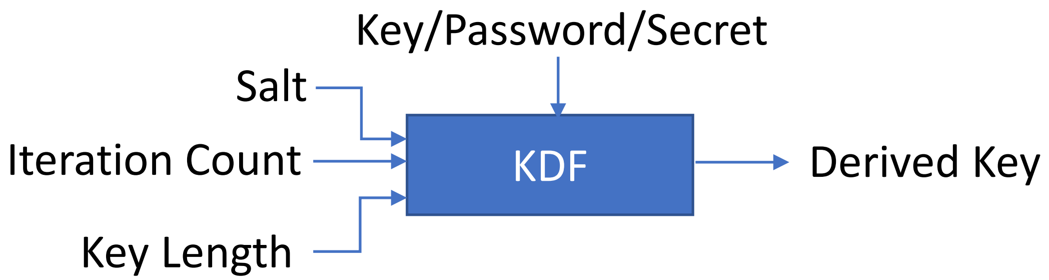
Cryptography for Software Engineers



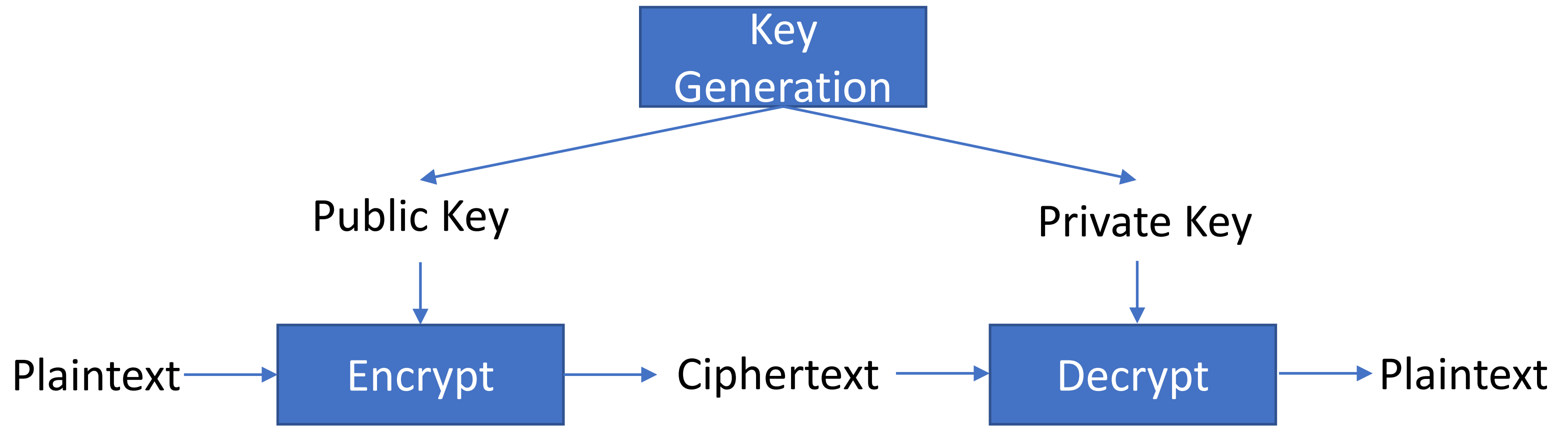


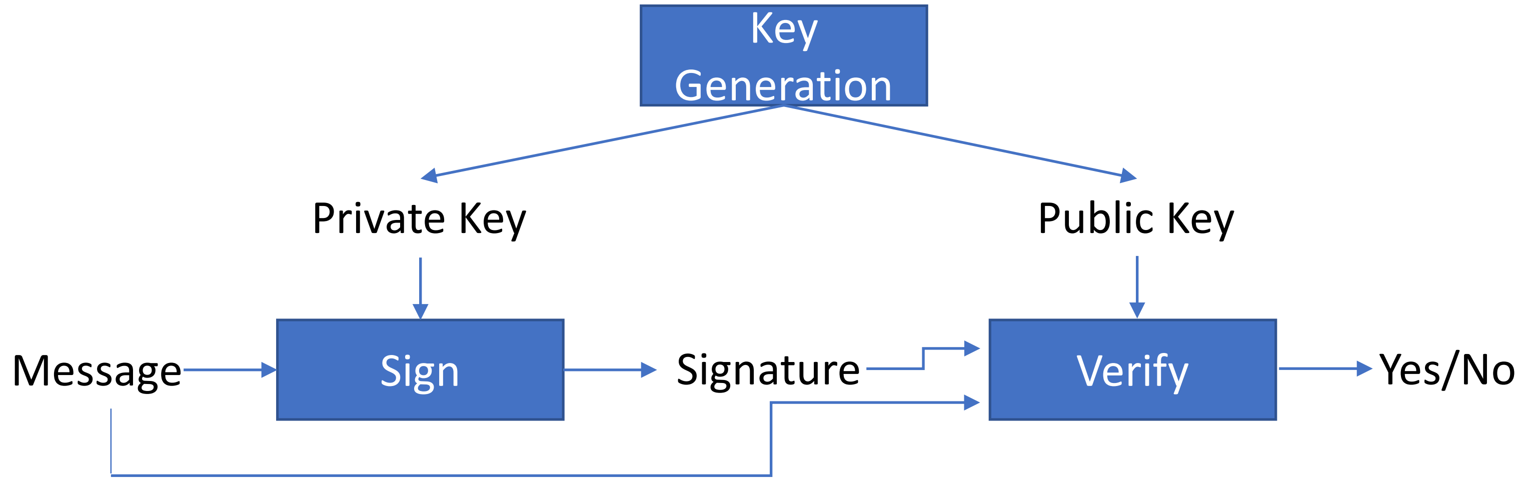




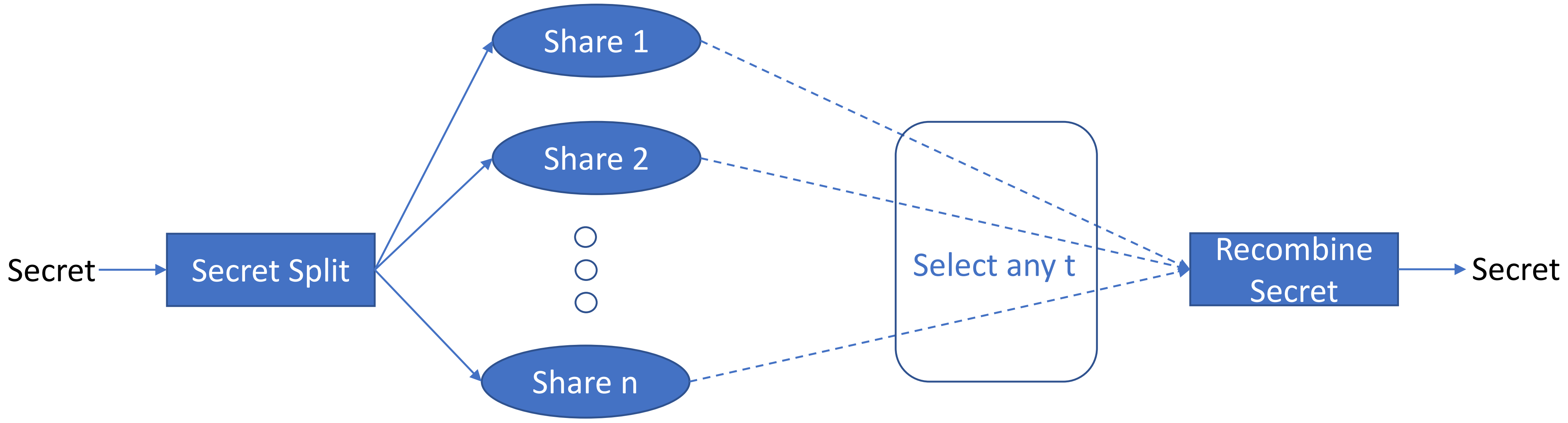


Cryptography for Software Engineers

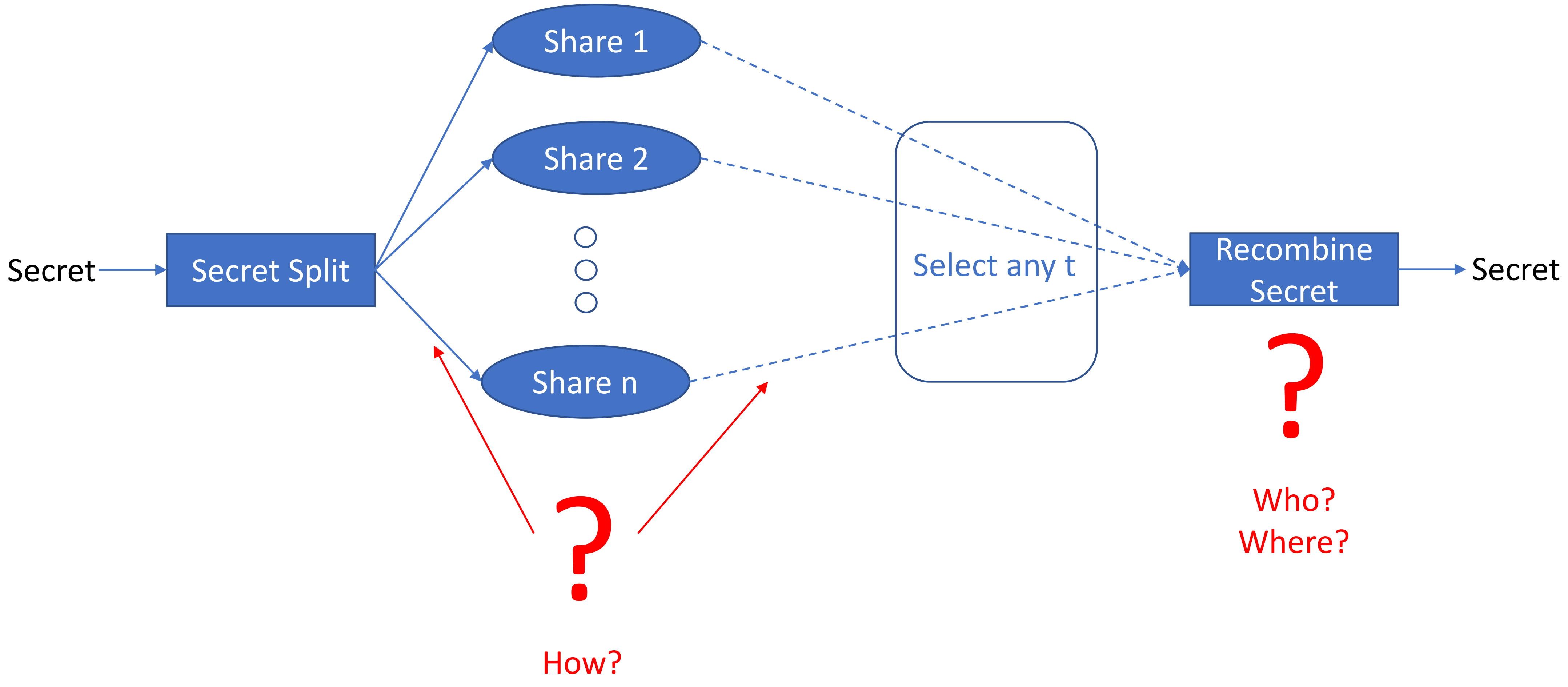




Threshold Cryptography for Software Engineers



Threshold Cryptography for Software Engineers



Conclusions

- General purpose threshold secret sharing, as a primitive, is simple to state but more complicated to use than other cryptographic primitives.
  - Notions of principals, channels and data is more ambiguous than with other primitives.
  - The gap between the definition of the algorithm and what software engineers want to do with it is wider than with other primitives.
  - General purpose formulation of threshold secret sharing is so abstract as to require too much work for non-cryptographic software engineers to use.
  - Specific formulation of threshold may not work in other scenarios.
- Software APIs closely match algorithm standards. Threshold secret sharing is almost too easy to state to even standardize.
  - However, there is room to provide guidance on how to use it (like SP800-130 Key Management.)

Conclusions

- Catch 22: General purpose threshold secret sharing is unlikely to be used without being standardized. Until it is widely used, we won’t know how to close the gap between use cases and specification.
- General purpose software engineers do not want a lot of ambiguity and flexibility with their cryptography algorithms.

Considerations for Standardization of Threshold Cryptography

- Post quantum competition affects the standardization of threshold cryptography.
  - If post quantum algorithms are available, why standardize a non-quantum safe threshold asymmetric algorithm?
  - If we are standardizing threshold schemes that are non-quantum secure, they have a limited window of utility.
- Cloud scenarios, increased government surveillance and crypto currencies all provide compelling and urgent use case for threshold crypto.
- There is urgency to standardize this, and an implicit deadline for when this would be worth doing.

Recommendations for Standardizing Threshold Cryptography

- Secret sharing (Shamir Secret Sharing) **IS NOT** ready for general purpose consumption.
  - NIST could provide guidance on how to use this. Pick the few most popular scenarios. For example, secrets stored and recombined in specialized hardware (HSM, TPM, Smart Cards); Secrets distributed over network recombined in software.
  - More research is needed, and it appears to be of limited value (to general consumers.)
  - Can be standardized as a component for use in *other* cryptographic protocols.
- ECDSA threshold signatures **ARE** very close to being ready for general consumption.
  - Use cases exist now.

Recommendations for Standardizing Threshold Cryptography

- The topic of threshold cryptography is a good candidate for standardization.
  - This should be done soon to satisfy current use cases and provide a useful lifetime before post quantum algorithms are mandated.
  - It does not necessarily need its own special publication, would it be more expedient to include this in a revision of other publications? (i.e. Can threshold ECDSA be included in revisions to ECDSA standards?)
  - If we can’t standardize this soon, we should wait until post quantum threshold schemes are ready.
  - Standardize algorithms that are to be used by general purpose software engineers, not just cryptographers.

Questions?