Towards Standardization of Threshold Schemes at NIST

Luís Brandão

Cryptographic Technology Group
National Institute of Standards and Technology
(Gaithersburg, Maryland, USA)

Presentation at the
Theory of Implementation Security (TIS'19) Workshop
November 11, 2019 @ London, UK

Some slides are based on previous presentations (NTCW'19; ICMC'19; ACS'19).

The NIST Threshold Cryptography project, on which this presentation is based, has so far also counted with the participation of Apostol Vassilev, Michael Davidson, Nicky Mouha.

Outline

1. Crypto standards at NIST
2. Threshold intro
3. Threshold project
4. Threshold preliminary roadmap
5. Concluding remarks

Outline

1. Crypto standards at NIST
2. Threshold intro
3. Threshold project
4. Threshold preliminary roadmap
5. Concluding remarks

Goals of this presentation:

- Overview of the NIST standardization effort
- Present the new "preliminary roadmap" (NISTIR 8214A)
- Encourage feedback and collaboration

Outline 1

1. Crypto standards at NIST
2. Threshold intro
3. Threshold project
4. Threshold preliminary roadmap
5. Concluding remarks

Some NIST data

National Institute of Standards and Technology (NIST)

(National Bureau of Standards 1901–1988 → NIST 1988–present)

- Non-regulatory federal agency (within the U.S. Department of Commerce)
- **Mission** (keywords): innovation, industrial competitiveness, measurement science, standards and technology, economic security, quality of life.

Aerial photo of Gaithersburg campus (source: Google Maps, August 2019)

Some NIST data

National Institute of Standards and Technology (NIST)

(National Bureau of Standards 1901–1988 $\to$ NIST 1988–present)

- Non-regulatory federal agency (within the U.S. Department of Commerce)
- **Mission** (keywords): innovation, industrial competitiveness, measurement science, standards and technology, economic security, quality of life.

Wide spectrum of competences

- $\sim 6\text{–}7 \times 10^3$ workers
- Five laboratories and two centers
- Laboratories $\to$ Divisions $\to$ Groups $\to$ Projects
- Standards, research and applications

Aerial photo of Gaithersburg campus (source: Google Maps, August 2019)

Laboratories, divisions, groups

Information Technology Laboratory (ITL):

advancing measurement science, standards, and technology through research and development in information technology, mathematics, and statistics.

Laboratories, divisions, groups

Information Technology Laboratory (ITL):

advancing measurement science, standards, and technology through research and development in information technology, mathematics, and statistics.

→ **Computer Security Division (CSD):** Cryptographic Technology; Secure Systems and Applications; Security Components and Mechanisms; Security Engineering and Risk Management; Security Testing, Validation and Measurement.

Laboratories, divisions, groups

Information Technology Laboratory (ITL):

advancing measurement science, standards, and technology through research and development in information technology, mathematics, and statistics.

- → **Computer Security Division (CSD):** Cryptographic Technology; Secure Systems and Applications; Security Components and Mechanisms; Security Engineering and Risk Management; Security Testing, Validation and Measurement.
  - → **Cryptographic Technology Group (CTG):** research, develop, engineer, and produce guidelines, recommendations and best practices for cryptographic algorithms, methods, and protocols.
  - → **Security Testing, Validation and Measurement (STVM):** validate cryptographic algorithm implementations, cryptographic modules, [...] develop test suites and test methods; provide implementation guidance [...]

Laboratories, divisions, groups

Information Technology Laboratory (ITL):

advancing measurement science, standards, and technology through research and development in information technology, mathematics, and statistics.

→ **Computer Security Division (CSD):** Cryptographic Technology; Secure Systems and Applications; Security Components and Mechanisms; Security Engineering and Risk Management; Security Testing, Validation and Measurement.

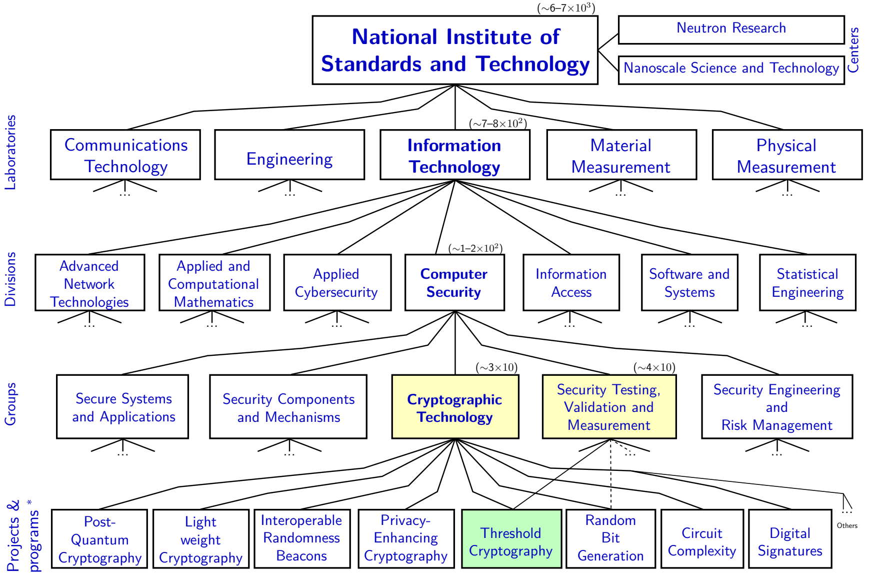
→ **Cryptographic Technology Group (CTG):** research, develop, engineer, and produce guidelines, recommendations and best practices for cryptographic algorithms, methods, and protocols.

→ **Security Testing, Validation and Measurement (STVM):** validate cryptographic algorithm implementations, cryptographic modules, [...] develop test suites and test methods; provide implementation guidance [...]

- Documents: FIPS, SP 800, NISTIR.
- International cooperation: government, industry, academia, standardization bodies.

FIPS = Federal Information Processing Standards; SP 800 = Special Publications in Computer Security; NISTIR = NIST Internal or Interagency Report.

Some projects of crypto primitives/applications at NIST



* (Some projects/programs involve several groups, divisions or labs) (in parenthesis: approximate range # workers, inc. associates and fed. employees)

Some standardized cryptographic primitives

Traditional focus on "basic" primitives:

Some standardized cryptographic primitives

Traditional focus on "basic" primitives:

- Block ciphers
- Cipher modes of operation
- Hash functions
- Signatures
- Pair-wise key agreement
- DRBGs

Some standardized cryptographic primitives

Traditional focus on "basic" primitives:

- Block ciphers: DES (1977), EES (1994), TDEA (1999), AES (2001)
- Cipher modes of operation (1980–): CBC, CT, CCM, GCM ...
- Hash functions (SHS): SHA-1 (1994), SHA-2 (2001), SHA-3 (2015)
- Signatures (DSS): DSA (1997), ECDSA (1998), RSA (2000), EdDSA (2019)
- Pair-wise key agreement, e.g., based on DH (2006) and RSA (2009)
- DRBGs (2006): CTR_, Hash_, HMAC_, Dual_EC_ (withdrawn in 2015 due to concerns of potential subversion)

(Not an exhaustive list; years indicated for perspective; some documentation has subsequent updates)
(Further details in "NIST Cryptographic Standards and Guidelines Development Program Briefing Book")

Legend:
- AES = Advanced Encryption Standard
- CBC = Cipher block chaining (mode)
- CT = Counter (mode)
- CCM = Counter with Cipher-block chaining
- DES = Data Encryption Standard
- DH = Diffie–Hellman
- DSA = Digital Signature Algorithm
- DSS = Digital Signature Standard
- DRBG = Deterministic Random Bit Generator
- ECDSA = Elliptic curve DSA
- EdDSA = Edwards curve DSA
- EES = Escrowed Encryption Standard
- GCM = Galois counter mode
- RSA = Rivest–Shamir–Adleman
- SHA = Secure Hash Algorithm
- SHS = Secure Hash Standard
- TDEA = Triple Data Encryption Algorithm

Some standardized cryptographic primitives

Traditional focus on "basic" primitives:

- Block ciphers: DES (1977), EES (1994), TDEA (1999), AES (2001)
- Cipher modes of operation (1980–): CBC, CT, CCM, GCM ...
- Hash functions (SHS): SHA-1 (1994), SHA-2 (2001), SHA-3 (2015)
- Signatures (DSS): DSA (1997), ECDSA (1998), RSA (2000), EdDSA (2019)
- Pair-wise key agreement, e.g., based on DH (2006) and RSA (2009)
- DRBGs (2006): CTR_, Hash_, HMAC_, Dual_EC_ (withdrawn in 2015 due to concerns of potential subversion)

(Not an exhaustive list; years indicated for perspective; some documentation has subsequent updates)
(Further details in "NIST Cryptographic Standards and Guidelines Development Program Briefing Book")

Legend:
- AES = Advanced Encryption Standard
- CBC = Cipher block chaining (mode)
- CT = Counter (mode)
- CCM = Counter with Cipher-block chaining
- DES = Data Encryption Standard
- DH = Diffie–Hellman
- DSA = Digital Signature Algorithm
- DSS = Digital Signature Standard
- DRBG = Deterministic Random Bit Generator
- ECDSA = Elliptic curve DSA
- EdDSA = Edwards curve DSA
- EES = Escrowed Encryption Standard
- GCM = Galois counter mode
- RSA = Rivest–Shamir–Adleman
- SHA = Secure Hash Algorithm
- SHS = Secure Hash Standard
- TDEA = Triple Data Encryption Algorithm

Some standardized cryptographic primitives

Traditional focus on "basic" primitives:

- Block ciphers: DES (1977), EES (1994), TDEA (1999), AES (2001)
- Cipher modes of operation (1980–): CBC, CT, CCM, GCM ...
- Hash functions (SHS): SHA-1 (1994), SHA-2 (2001), SHA-3 (2015)
- Signatures (DSS): DSA (1997), ECDSA (1998), RSA (2000), EdDSA (2019)
- Pair-wise key agreement, e.g., based on DH (2006) and RSA (2009)
- DRBGs (2006): CTR_, Hash_, HMAC_, Dual_EC_ (withdrawn in 2015 due to concerns of potential subversion)

(Not an exhaustive list; years indicated for perspective; some documentation has subsequent updates)
(Further details in "NIST Cryptographic Standards and Guidelines Development Program Briefing Book")

Legend:
- AES = Advanced Encryption Standard
- CBC = Cipher block chaining (mode)
- CT = Counter (mode)
- CCM = Counter with Cipher-block chaining
- DES = Data Encryption Standard
- DH = Diffie–Hellman
- DSA = Digital Signature Algorithm
- DSS = Digital Signature Standard
- DRBG = Deterministic Random Bit Generator
- ECDSA = Elliptic curve DSA
- EdDSA = Edwards curve DSA
- EES = Escrowed Encryption Standard
- GCM = Galois counter mode
- RSA = Rivest–Shamir–Adleman
- SHA = Secure Hash Algorithm
- SHS = Secure Hash Standard
- TDEA = Triple Data Encryption Algorithm

Some standardized cryptographic primitives

Traditional focus on "basic" primitives:

- Block ciphers: DES (1977), EES (1994), TDEA (1999), AES (2001)
- Cipher modes of operation (1980–): CBC, CT, CCM, GCM ...
- Hash functions (SHS): SHA-1 (1994), SHA-2 (2001), SHA-3 (2015)
- Signatures (DSS): DSA (1997), ECDSA (1998), RSA (2000), EdDSA (2019)
- Pair-wise key agreement, e.g., based on DH (2006) and RSA (2009)
- DRBGs (2006): CTR_, Hash_, HMAC_, Dual_EC_
  (withdrawn in 2015 due to concerns of potential subversion)

(Not an exhaustive list; years indicated for perspective; some documentation has subsequent updates)
(Further details in "NIST Cryptographic Standards and Guidelines Development Program Briefing Book")

Legend:
- AES = Advanced Encryption Standard
- CBC = Cipher block chaining (mode)
- CT = Counter (mode)
- CCM = Counter with Cipher-block chaining
- DES = Data Encryption Standard
- DH = Diffie–Hellman
- DSA = Digital Signature Algorithm
- DSS = Digital Signature Standard
- DRBG = Deterministic Random Bit Generator
- ECDSA = Elliptic curve DSA
- EdDSA = Edwards curve DSA
- EES = Escrowed Encryption Standard
- GCM = Galois counter mode
- RSA = Rivest–Shamir–Adleman
- SHA = Secure Hash Algorithm
- SHS = Secure Hash Standard
- TDEA = Triple Data Encryption Algorithm

Some standardized cryptographic primitives

Traditional focus on “basic” primitives:

- Block ciphers: DES (1977), EES (1994), TDEA (1999), AES (2001)
- Cipher modes of operation (1980–): CBC, CT, CCM, GCM ...
- Hash functions (SHS): SHA-1 (1994), SHA-2 (2001), SHA-3 (2015)
- Signatures (DSS): DSA (1997), ECDSA (1998), RSA (2000), EdDSA (2019)
- Pair-wise key agreement, e.g., based on DH (2006) and RSA (2009)
- DRBGs (2006): CTR_, Hash_, HMAC_, Dual_EC_ (withdrawn in 2015 due to concerns of potential subversion)

(Not an exhaustive list; years indicated for perspective; some documentation has subsequent updates)
(Further details in “NIST Cryptographic Standards and Guidelines Development Program Briefing Book”)

Some of these NIST-standards were specified with reference to standards by other bodies, and with further requirements.

Legend:
- AES = Advanced Encryption Standard
- CBC = Cipher block chaining (mode)
- CT = Counter (mode)
- CCM = Counter with Cipher-block chaining
- DES = Data Encryption Standard
- DH = Diffie–Hellman
- DSA = Digital Signature Algorithm
- DSS = Digital Signature Standard
- DRBG = Deterministic Random Bit Generator
- ECDSA = Elliptic curve DSA
- EdDSA = Edwards curve DSA
- EES = Escrowed Encryption Standard
- GCM = Galois counter mode
- RSA = Rivest–Shamir–Adleman
- SHA = Secure Hash Algorithm
- SHS = Secure Hash Standard
- TDEA = Triple Data Encryption Algorithm

Some standardized cryptographic primitives

Traditional focus on "basic" primitives:

- Block ciphers: DES (1977), EES (1994), TDEA (1999), AES (2001)
- Cipher modes of operation (1980–): CBC, CT, CCM, GCM ...
- Hash functions (SHS): SHA-1 (1994), SHA-2 (2001), SHA-3 (2015)
- Signatures (DSS): DSA (1997), ECDSA (1998), RSA (2000), EdDSA (2019)
- Pair-wise key agreement, e.g., based on DH (2006) and RSA (2009)
- DRBGs (2006): CTR_, Hash_, HMAC_, Dual_EC_ (withdrawn in 2015 due to concerns of potential subversion)

(Not an exhaustive list; years indicated for perspective; some documentation has subsequent updates)
(Further details in "NIST Cryptographic Standards and Guidelines Development Program Briefing Book")

Legend:
- AES = Advanced Encryption Standard
- CBC = Cipher block chaining (mode)
- CT = Counter (mode)
- CCM = Counter with Cipher-block chaining
- DES = Data Encryption Standard
- DH = Diffie–Hellman
- DSA = Digital Signature Algorithm
- DSS = Digital Signature Standard
- DRBG = Deterministic Random Bit Generator
- ECDSA = Elliptic curve DSA
- EdDSA = Edwards curve DSA
- EES = Escrowed Encryption Standard
- GCM = Galois counter mode
- RSA = Rivest–Shamir–Adleman
- SHA = Secure Hash Algorithm
- SHS = Secure Hash Standard
- TDEA = Triple Data Encryption Algorithm

Some of these NIST-standards were specified with reference to standards by other bodies, and with further requirements.

Several methods:

- Internal or interagency developed techniques
- Adoption of external standards
- Open call, competition, "competition-like"

Other processes (examples)

Other processes (examples)

Ongoing evaluations:

- Post-quantum cryptography: signatures, public-key encryption, key encapsulation
- Lightweight cryptography: ciphers, authenticated encryption, hash functions

Other processes (examples)

Ongoing evaluations:

- Post-quantum cryptography: signatures, public-key encryption, key encapsulation
- Lightweight cryptography: ciphers, authenticated encryption, hash functions

The crypto group has other ongoing projects: https://www.nist.gov/itl/csd/cryptographic-technology

Other processes (examples)

Ongoing evaluations:

- Post-quantum cryptography: signatures, public-key encryption, key encapsulation
- Lightweight cryptography: ciphers, authenticated encryption, hash functions

The crypto group has other ongoing projects: https://www.nist.gov/itl/csd/cryptographic-technology

Previous considerations:

- Paring-based Cryptography: workshop (2008), study and call for feedback on use cases (2011), report (2012–2015) (forming NIST's position on standardization/recommendation: more research is needed).

Other processes (examples)

Ongoing evaluations:

- Post-quantum cryptography: signatures, public-key encryption, key encapsulation
- Lightweight cryptography: ciphers, authenticated encryption, hash functions

The crypto group has other ongoing projects: https://www.nist.gov/itl/csd/cryptographic-technology

Previous considerations:

- Paring-based Cryptography: workshop (2008), study and call for feedback on use cases (2011), report (2012–2015) (forming NIST's position on standardization/recommendation: more research is needed).

Development process:

- NISTIR 7977: NIST Cryptographic Standards and Guidelines Development Process (2016). Formalizes several **principles** to follow:
  - transparency
  - openness
  - balance
  - integrity
  - technical merit
  - usability
  - global acceptability
  - continuous improvement
  - innovation and intellectual property

  (and overarching considerations)

Outline 2

1. Crypto standards at NIST
2. Threshold intro
3. Threshold project
4. Threshold preliminary roadmap
5. Concluding remarks

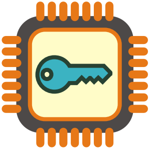
Beyond defining basic crypto primitives?

Security often hinges on a good application of cryptography

Beyond defining basic crypto primitives?

Security often hinges on a good application of cryptography

Specially relevant: key-based cryptographic primitives

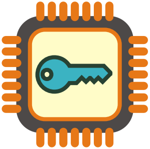


Beyond defining basic crypto primitives?

Security often hinges on a good application of cryptography

Specially relevant: **key**-based cryptographic primitives

Security relies on:



openclipart.org/detail/101407

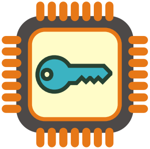
- secrecy, correctness, availability ... of cryptographic **keys**

Beyond defining basic crypto primitives?

Security often hinges on a good application of cryptography

Specially relevant: **key**-based cryptographic primitives

Security relies on:



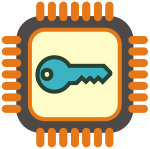
- secrecy, correctness, availability ... of cryptographic **keys**
- **implementations** that use **keys** to operate an algorithm

Beyond defining basic crypto primitives?

Security often hinges on a good application of cryptography

Specially relevant: **key**-based cryptographic primitives

Security relies on:



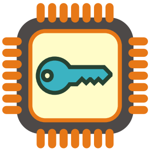
- secrecy, correctness, availability ... of cryptographic **keys**
- **implementations** that use **keys** to operate an algorithm
- **operators** to decide when/where to apply the algorithms

Beyond defining basic crypto primitives?

Security often hinges on a good application of cryptography

Specially relevant: **key**-based cryptographic primitives

Security relies on:



- secrecy, correctness, availability ... of cryptographic **keys**
- **implementations** that use **keys** to operate an algorithm
- **operators** to decide when/where to apply the algorithms

Some things can go wrong!

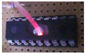
Crypto can be affected by vulnerabilities!

Crypto can be affected by vulnerabilities!

Attacks can exploit differences between ideal vs. real **implementations**

Crypto can be affected by vulnerabilities!

Attacks can exploit differences between ideal vs. real **implementations**

"Bellcore attack" (1997)
[BDL97]

[SH07]

Cold-boot attacks (2009)
[HSH+09]

[Don13]

Heartbleed bug (2014)
[DLK+14]
heartbleed.com

"ZigBee Chain reaction" (2017)
[RSWO17]

[RSWO17]

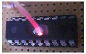
Meltdown & Spectre (2017)
[LSG+18, KGG+18]
meltdownattack.com

Foreshadow (2018)
[BMW+18, WBM+18]
foreshadowattack.eu

Microarchitectural Data Sampling (2019)
[MDS19]
mdsattacks.com

Crypto can be affected by vulnerabilities!

Attacks can exploit differences between ideal vs. real **implementations**

"Bellcore attack" (1997)
[BDL97]

[SH07]

Cold-boot attacks (2009)
[HSH+09]

[Don13]

Heartbleed bug (2014)
[DLK+14]
heartbleed.com

"ZigBee Chain reaction" (2017)
[RSWO17]

[RSWO17]

Meltdown & Spectre (2017)
[LSG+18, KGG+18]
meltdownattack.com

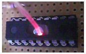
Foreshadow (2018)
[BMW+18, WBM+18]
foreshadowattack.eu

Microarchitectural Data Sampling (2019)
[MDS19]
mdsattacks.com

Operators of cryptographic implementations can go rogue

Crypto can be affected by vulnerabilities!

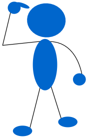
Attacks can exploit differences between ideal vs. real **implementations**

“Bellcore attack” (1997)	Cold-boot attacks (2009)	Heartbleed bug (2014)	“ZigBee Chain reaction” (2017)	Meltdown & Spectre (2017)	Foreshadow (2018)	Microarchitectural Data Sampling (2019)
[BDL97]	[HSH+09]	[DLK+14]	[RSWO17]	[LSG+18, KGG+18]	[BMW+18, WBM+18]	[MDS19]
 [SH07]	 [Don13]	heartbleed.com	 [RSWO17]	meltdownattack.com	foreshadowattack.eu	mdsattacks.com

Operators of cryptographic implementations can go rogue

How can we address single-points of failure?

*question-2.html



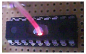
*4296.html

* = clker.com/clipart-

Crypto can be affected by vulnerabilities!

Attacks can exploit differences between ideal vs. real **implementations**

"Bellcore attack" (1997) [BDL97]



[SH07]

Cold-boot attacks (2009) [HSH+09]

[Don13]

Heartbleed bug (2014) [DLK+14]

heartbleed.com

"ZigBee Chain reaction" (2017) [RSWO17]

[RSWO17]

Meltdown & Spectre (2017) [LSG+18, KGG+18]

meltdownattack.com

Foreshadow (2018) [BMW+18, WBM+18]

foreshadowattack.eu

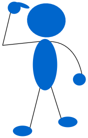
Microarchitectural Data Sampling (2019) [MDS19]

mdsattacks.com

Operators of cryptographic implementations can go rogue

How can we address single-points of failure?

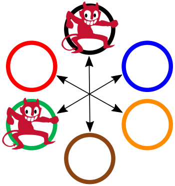
*question-2.html



*4296.html

* = clker.com/clipart-

The threshold approach

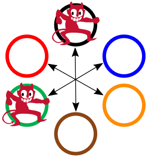


The red dancing devil is from clker.com/clipart-13643.html

The threshold approach

At a high-level:

use redundancy & diversity to mitigate the *compromise* of up to a threshold number (f-out-of-n) of components

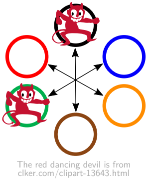


The red dancing devil is from clker.com/clipart-13643.html

The threshold approach

At a high-level:

use redundancy & diversity to mitigate the *compromise* of up to a threshold number (f-out-of-n) of components



The intuitive aim:

improve security

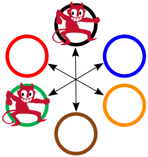
vs.

a non-threshold scheme

The threshold approach

At a high-level:

use redundancy & diversity to mitigate the *compromise* of up to a threshold number (f-out-of-n) of components



The red dancing devil is from clker.com/clipart-13643.html

The intuitive aim:

improve security

vs.

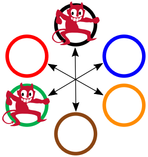
a non-threshold scheme

NIST-CSD wants to standardize
threshold schemes for cryptographic primitives

The threshold approach

At a high-level:

use redundancy & diversity to mitigate the *compromise* of up to a threshold number (f-out-of-n) of components



The red dancing devil is from clker.com/clipart-13643.html

The intuitive aim:

improve security

vs.

a non-threshold scheme

NIST-CSD wants to standardize
threshold schemes for cryptographic primitives

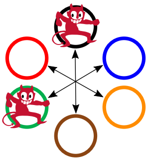
Potential primitives: sign, decrypt (PKE), encipher/decipher, key generate, ...

(PKE) = within a **p**ublic-**k**ey **e**ncryption scheme

The threshold approach

At a high-level:

use redundancy & diversity to mitigate the *compromise* of up to a threshold number (f-out-of-n) of components



The red dancing devil is from clker.com/clipart-13643.html

The intuitive aim:

improve security vs. a non-threshold scheme

NIST-CSD wants to standardize
threshold schemes for cryptographic primitives

Potential primitives: sign, decrypt (PKE), encipher/decipher, key generate, ...

(PKE) = within a **p**ublic-**k**ey **e**ncryption scheme

Some properties:

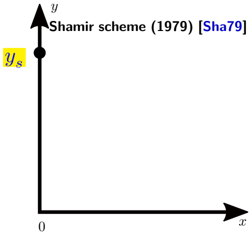
- **withstands** several *compromised* components;
- **needs** several *uncompromised* components;
- **prevents** secret keys from being in one place;
- **enhances** resistance against side-channel attacks; ...

Secret Sharing Schemes (a starting point)

Split a secret key into n secret "shares" for storage at rest.

Secret Sharing Schemes (a starting point)

Split a secret key into n secret "shares" for storage at rest.

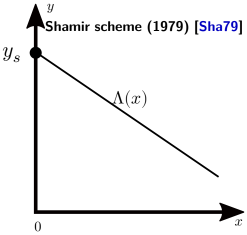


Example 2-out-of-*n* secret sharing

- The secret y_s is placed in the y-axis;

Secret Sharing Schemes (a starting point)

Split a secret key into n secret "shares" for storage at rest.

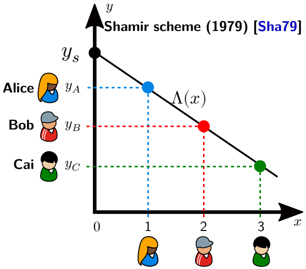


Example 2-out-of-*n* secret sharing

- The secret y_s is placed in the y-axis;
- A random line Λ is drawn crossing the secret;

Secret Sharing Schemes (a starting point)

Split a secret key into n secret "shares" for storage at rest.

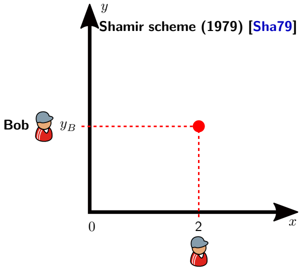


Example 2-out-of-*n* secret sharing

- The secret y_s is placed in the y-axis;
- A random line Λ is drawn crossing the secret;
- Each share is a point $(\Lambda(i), i)$ in the line Λ;

Secret Sharing Schemes (a starting point)

Split a secret key into n secret "shares" for storage at rest.



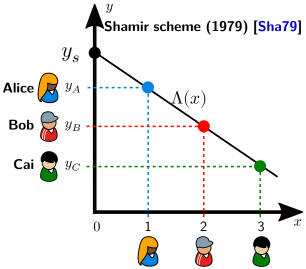
Example 2-out-of-n secret sharing

- The secret y_s is placed in the y-axis;
- A random line Λ is drawn crossing the secret;
- Each share is a point $(\Lambda(i), i)$ in the line Λ;

Each share alone has no information about the secret.

Secret Sharing Schemes (a starting point)

Split a secret key into n secret “shares” for storage at rest.



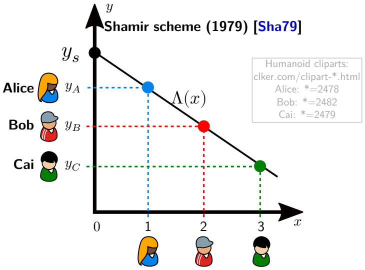
Example 2-out-of-*n* secret sharing

- The secret y_s is placed in the y-axis;
- A random line Λ is drawn crossing the secret;
- Each share is a point $(\Lambda(i), i)$ in the line Λ;

Each share alone has no information about the secret.
Any pair of shares allows recovering the secret.

Secret Sharing Schemes (a starting point)

Split a secret key into n secret "shares" for storage at rest.



Example 2-out-of-*n* secret sharing

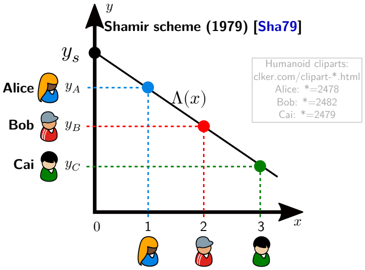
- The secret y_s is placed in the y-axis;
- A random line Λ is drawn crossing the secret;
- Each share is a point $(\Lambda(i), i)$ in the line Λ;

Each share alone has no information about the secret.
Any pair of shares allows recovering the secret.

But how to avoid recombining the key when the key is needed by an algorithm?

Secret Sharing Schemes (a starting point)

Split a secret key into n secret "shares" for storage at rest.



Example 2-out-of-*n* secret sharing

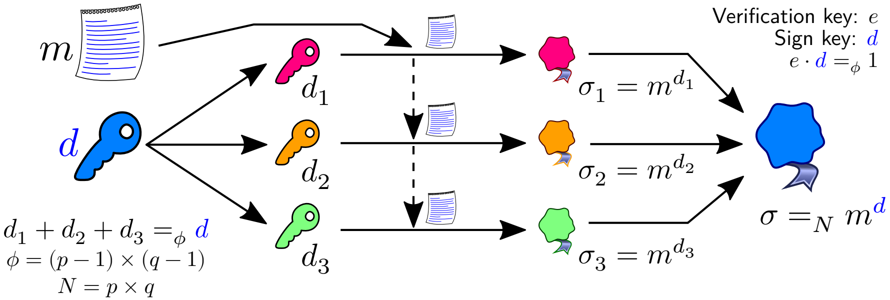
- The secret y_s is placed in the y-axis;
- A random line Λ is drawn crossing the secret;
- Each share is a point $(\Lambda(i), i)$ in the line Λ;

Each share alone has no information about the secret.
Any pair of shares allows recovering the secret.

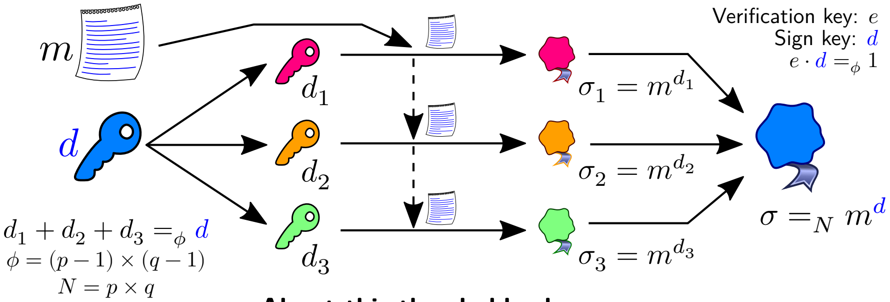
But how to avoid recombining the key when the key is needed by an algorithm?

Use threshold schemes for cryptographic primitives (next)

A simple example: RSA signature (or decryption) [RSA78]



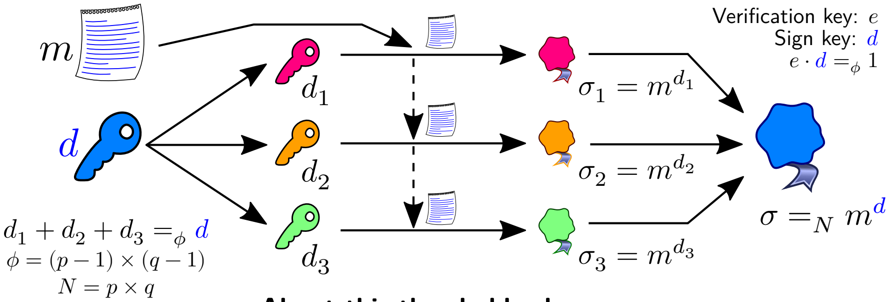
A simple example: RSA signature (or decryption) [RSA78]



About this threshold scheme:

SignKey d not recombined;

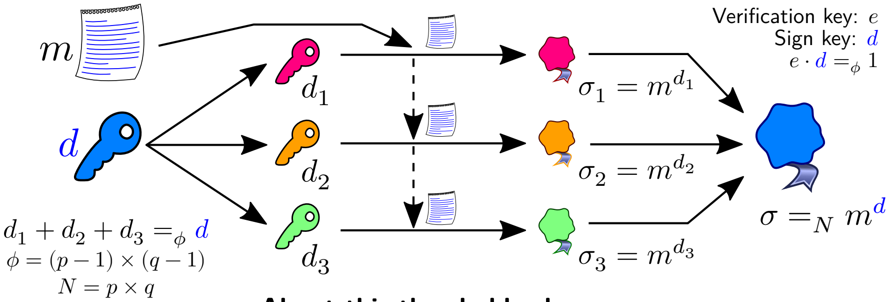
A simple example: RSA signature (or decryption) [RSA78]



About this threshold scheme:

SignKey d not recombined; can *reshare* d leaving e fixed;

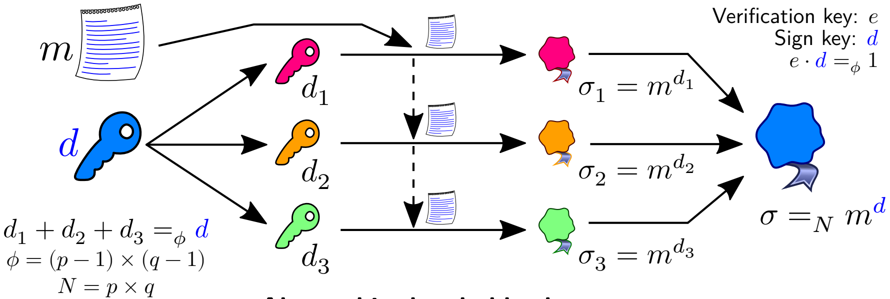
A simple example: RSA signature (or decryption) [RSA78]



About this threshold scheme:

SignKey d not recombined; can *reshare* d leaving e fixed; same σ;

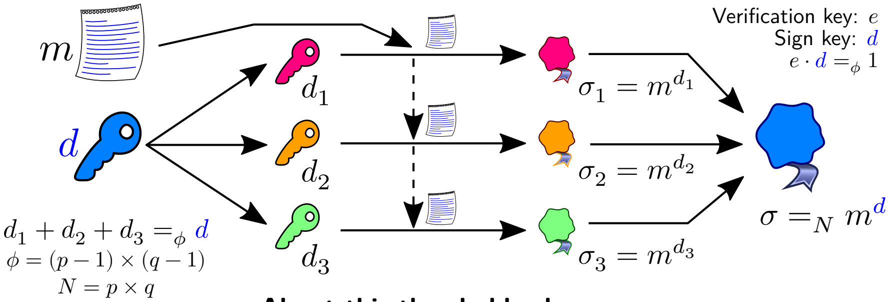
A simple example: RSA signature (or decryption) [RSA78]



About this threshold scheme:

SignKey d not recombined; can *reshare* d leaving e fixed; same σ; efficient!

A simple example: RSA signature (or decryption) [RSA78]

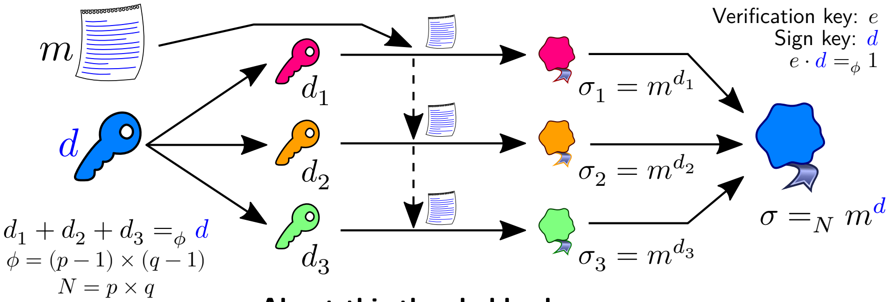


About this threshold scheme:

SignKey d not recombined; can *reshare* d leaving e fixed; same σ; efficient!

Facilitating setting: $\exists$ dealer;

A simple example: RSA signature (or decryption) [RSA78]

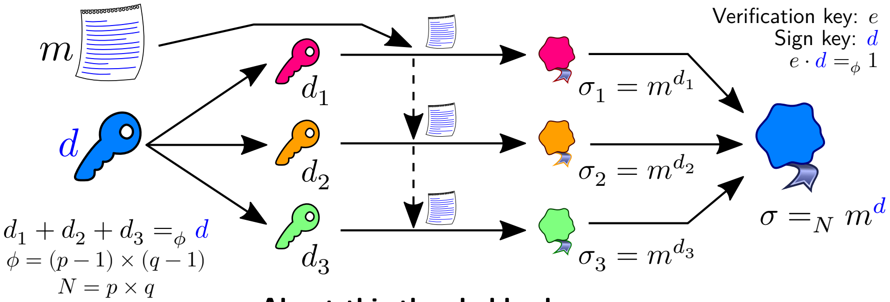


About this threshold scheme:

SignKey d not recombined; can *reshare* d leaving e fixed; same σ; efficient!

Facilitating setting: $\exists$ dealer; $\exists$ homomorphism;

A simple example: RSA signature (or decryption) [RSA78]

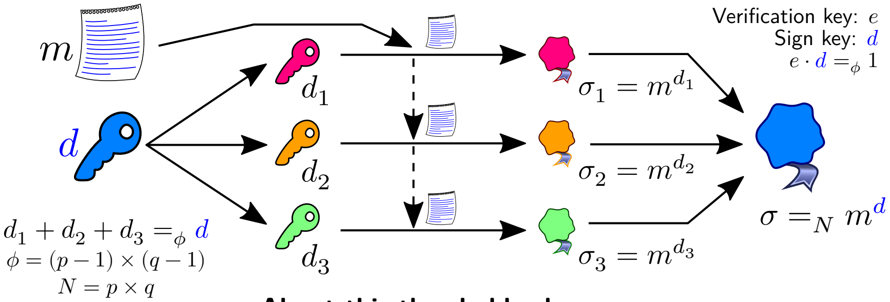


About this threshold scheme:

SignKey d not recombined; can *reshare* d leaving e fixed; same σ; efficient!

Facilitating setting: $\exists$ dealer; $\exists$ homomorphism; all parties learn m.

A simple example: RSA signature (or decryption) [RSA78]



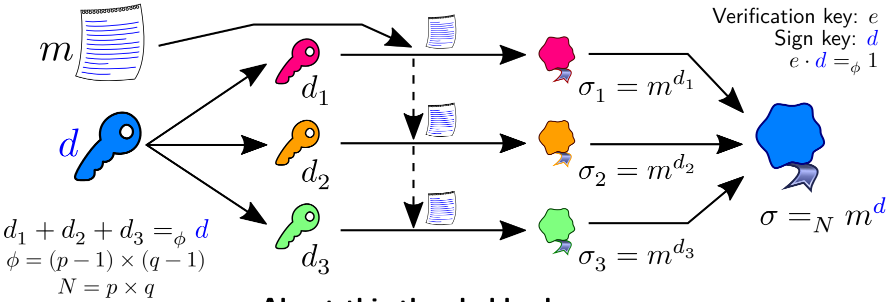
About this threshold scheme:

SignKey d not recombined; can *reshare* d leaving e fixed; same σ; efficient!

Facilitating setting: $\exists$ dealer; $\exists$ homomorphism; all parties learn m.

Not fault-tolerant: a single sub-signer can boycott a correct signing.

A simple example: RSA signature (or decryption) [RSA78]



About this threshold scheme:

SignKey d not recombined; can *reshare* d leaving e fixed; same σ; efficient!

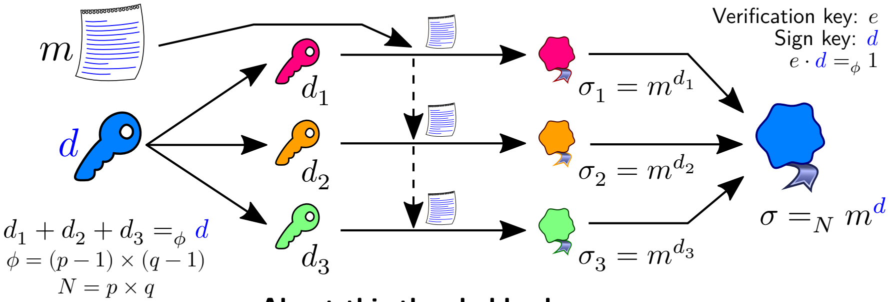
Facilitating setting: $\exists$ dealer; $\exists$ homomorphism; all parties learn m.

Not fault-tolerant: a single sub-signer can boycott a correct signing.

Can other threshold schemes be implemented:

?

A simple example: RSA signature (or decryption) [RSA78]



About this threshold scheme:

SignKey d not recombined; can *reshare* d leaving e fixed; same σ; efficient!

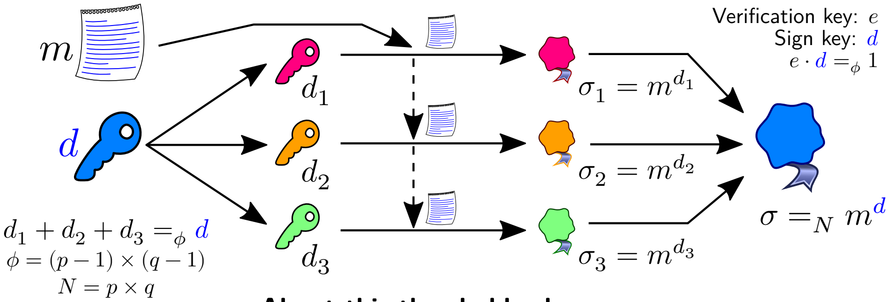
Facilitating setting: $\exists$ dealer; $\exists$ homomorphism; all parties learn m.

Not fault-tolerant: a single sub-signer can boycott a correct signing.

Can other threshold schemes be implemented:

$\nexists$ dealer, $\nexists$ homomorphisms, secret-shared m, withstanding f malicious signers?

A simple example: RSA signature (or decryption) [RSA78]



About this threshold scheme:

SignKey d not recombined; can *reshare* d leaving e fixed; same σ; efficient!

Facilitating setting: $\exists$ dealer; $\exists$ homomorphism; all parties learn m.

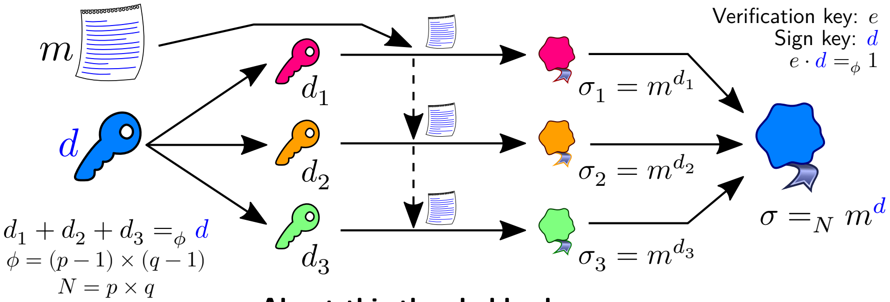
Not fault-tolerant: a single sub-signer can boycott a correct signing.

Can other threshold schemes be implemented:

$\nexists$ dealer, $\nexists$ homomorphisms, secret-shared m, withstanding f malicious signers?

Yes, using threshold cryptography

A simple example: RSA signature (or decryption) [RSA78]



About this threshold scheme:

SignKey d not recombined; can *reshare* d leaving e fixed; same σ; efficient!

Facilitating setting: $\exists$ dealer; $\exists$ homomorphism; all parties learn m.

Not fault-tolerant: a single sub-signer can boycott a correct signing.

Can other threshold schemes be implemented:

$\nexists$ dealer, $\nexists$ homomorphisms, secret-shared m, withstanding f malicious signers?

Yes, using threshold cryptography (with more complicated schemes)

What do the thresholds k and f mean?

What do the thresholds k and f mean?

3-out-of-3 decryption:

- **Availability:** 3 nodes needed to decrypt
- **Key secrecy:** okay while 1 share is secret

clker.com/clipart-encryption.html

What do the thresholds k and f mean?

3-out-of-3 decryption:

- **Availability:** 3 nodes needed to decrypt $(k = 3,\ f = 0)$
- **Key secrecy:** okay while 1 share is secret

clker.com/clipart-encryption.html

What do the thresholds k and f mean?

3-out-of-3 decryption:

- **Availability:** 3 nodes needed to decrypt ($k = 3,\ f = 0$)
- **Key secrecy:** okay while 1 share is secret ($k = 1,\ f = 2$)

clker.com/clipart-encryption.html

What do the thresholds k and f mean?

3-out-of-3 decryption:

- **Availability:** 3 nodes needed to decrypt ($k = 3,\ f = 0$)
- **Key secrecy:** okay while 1 share is secret ($k = 1,\ f = 2$)

clker.com/clipart-encryption.html

(Each security property has its own k and f)

What do the thresholds k and f mean?

3-out-of-3 decryption:

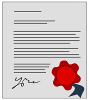
- **Availability:** 3 nodes needed to decrypt ($k = 3$, $f = 0$)
- **Key secrecy:** okay while 1 share is secret ($k = 1$, $f = 2$)

clker.com/clipart-encryption.html

(Each security property has its own k and f)

2-out-of-3 signature:

- **Availability:** 2 nodes needed to sign
- **Key secrecy:** okay while 2 shares are secret



clker.com/clipart-3712.html

What do the thresholds k and f mean?

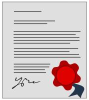
3-out-of-3 decryption:

clker.com/clipart-encryption.html

- **Availability:** 3 nodes needed to decrypt ($k = 3,\ f = 0$)
- **Key secrecy:** okay while 1 share is secret ($k = 1,\ f = 2$)

(Each security property has its own k and f)

2-out-of-3 signature:


clker.com/clipart-3712.html

- **Availability:** 2 nodes needed to sign ($k = 2, f = 1$)
- **Key secrecy:** okay while 2 shares are secret

What do the thresholds k and f mean?

3-out-of-3 decryption:

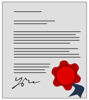
- **Availability:** 3 nodes needed to decrypt ($k = 3,\ f = 0$)
- **Key secrecy:** okay while 1 share is secret ($k = 1,\ f = 2$)

clker.com/clipart-encryption.html

(Each security property has its own k and f)

2-out-of-3 signature:

- **Availability:** 2 nodes needed to sign ($k = 2, f = 1$)
- **Key secrecy:** okay while 2 shares are secret ($k = 2, f = 1$)



clker.com/clipart-3712.html

What do the thresholds k and f mean?

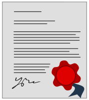
3-out-of-3 decryption:

clker.com/clipart-encryption.html

- **Availability:** 3 nodes needed to decrypt ($k = 3,\ f = 0$)
- **Key secrecy:** okay while 1 share is secret ($k = 1,\ f = 2$)

(Each security property has its own k and f)

2-out-of-3 signature:


clker.com/clipart-3712.html

- **Availability:** 2 nodes needed to sign ($k = 2, f = 1$)
- **Key secrecy:** okay while 2 shares are secret ($k = 2, f = 1$)

But does any of these schemes improve security?

(compared with a non-threshold scheme ($n = k = 1, f = 0$))

What do the thresholds k and f mean?

3-out-of-3 decryption:

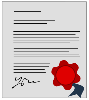
- **Availability:** 3 nodes needed to decrypt ($k = 3,\ f = 0$)
- **Key secrecy:** okay while 1 share is secret ($k = 1,\ f = 2$)

clker.com/clipart-encryption.html

(Each security property has its own k and f)

2-out-of-3 signature:

- **Availability:** 2 nodes needed to sign ($k = 2, f = 1$)
- **Key secrecy:** okay while 2 shares are secret ($k = 2, f = 1$)



clker.com/clipart-3712.html

But does any of these schemes improve security?

(compared with a non-threshold scheme ($n = k = 1, f = 0$))

It depends: "k-out-of-n" or "f-out-of-n" is not a sufficient characterization for a comprehensive security assertion

What do the thresholds k and f mean?

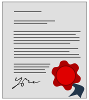
3-out-of-3 decryption:

clker.com/clipart-encryption.html

- **Availability:** 3 nodes needed to decrypt ($k = 3,\ f = 0$)
- **Key secrecy:** okay while 1 share is secret ($k = 1,\ f = 2$)

(Each security property has its own k and f)

2-out-of-3 signature:



clker.com/clipart-3712.html

- **Availability:** 2 nodes needed to sign ($k = 2, f = 1$)
- **Key secrecy:** okay while 2 shares are secret ($k = 2, f = 1$)

But does any of these schemes improve security?

(compared with a non-threshold scheme ($n = k = 1, f = 0$))

It depends: "k-out-of-n" or "f-out-of-n" is not a sufficient characterization for a comprehensive security assertion

Depends on attack model (e.g., attack surface, ...), system model (e.g., rejuvenations, ...), ...

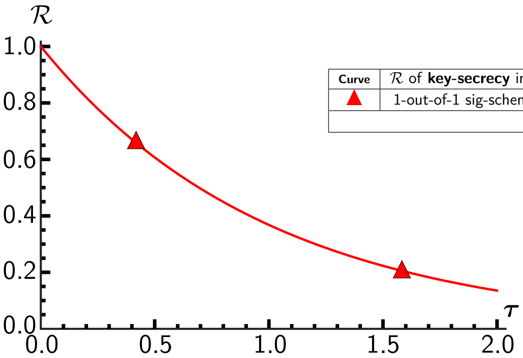
Reliability ($\mathcal{R}$) as one metric of security

Probability that a security property (e.g., secrecy) never fails during a mission time

Reliability ($\mathcal{R}$) as one metric of security

Probability that a security property (e.g., secrecy) never fails during a mission time

A possible model: each node fails (independently) with constant rate probability



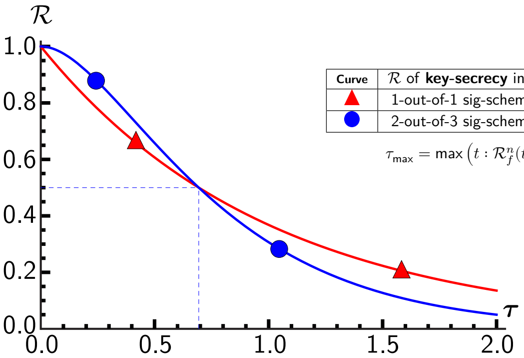
Curve	$\mathcal{R}$ of **key-secrecy** in a	n	f	τ_{max}
▲	1-out-of-1 sig-scheme	1	0	—

[BB12] Time normalized: $\tau = 1$ is the expected time to failure (ETTF) of a node

Reliability ($\mathcal{R}$) as one metric of security

Probability that a security property (e.g., secrecy) never fails during a mission time

A possible model: each node fails (independently) with constant rate probability



Curve	$\mathcal{R}$ of **key-secrecy** in a	n	f	τ_{max}
▲	1-out-of-1 sig-scheme	1	0	—
●	2-out-of-3 sig-scheme	3	1	0.693

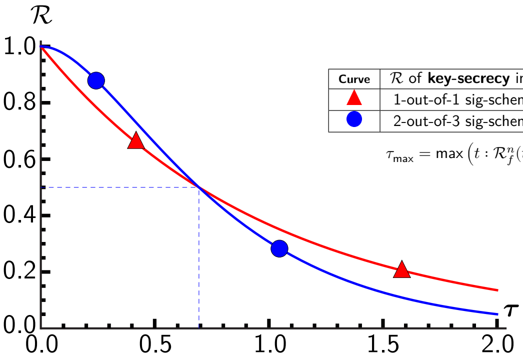
$$\tau_{\text{max}} = \max\left(t : \mathcal{R}_f^n(t) > \mathcal{R}_0^1(t)\right)$$

[BB12] Time normalized: $\tau = 1$ is the expected time to failure (ETTF) of a node

Reliability ($\mathcal{R}$) as one metric of security

Probability that a security property (e.g., secrecy) never fails during a mission time

A possible model: each node fails (independently) with constant rate probability



Curve	$\mathcal{R}$ of **key-secrecy** in a	n	f	τ_{max}
▲	1-out-of-1 sig-scheme	1	0	—
●	2-out-of-3 sig-scheme	3	1	0.693

$\tau_{\text{max}} = \max\left(t : \mathcal{R}^n_f(t) > \mathcal{R}^1_0(t)\right)$

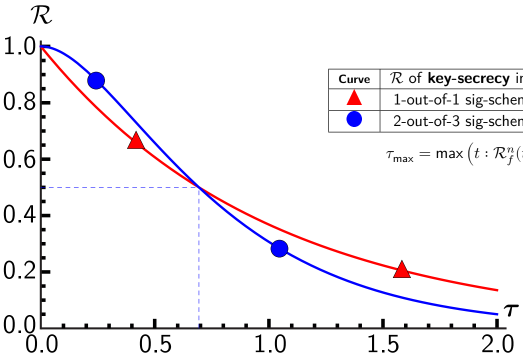
[BB12] Time normalized: $\tau = 1$ is the expected time to failure (ETTF) of a node

Increasing the fault-tolerance threshold f may degrade *reliability*

Reliability ($\mathcal{R}$) as one metric of security

Probability that a security property (e.g., secrecy) never fails during a mission time

A possible model: each node fails (independently) with constant rate probability



Curve	$\mathcal{R}$ of **key-secrecy** in a	n	f	τ_{max}
▲	1-out-of-1 sig-scheme	1	0	—
●	2-out-of-3 sig-scheme	3	1	0.693

$\tau_{\text{max}} = \max\left(t : \mathcal{R}_f^n(t) > \mathcal{R}_0^1(t)\right)$

[BB12] Time normalized: $\tau = 1$ is the expected time to failure (ETTF) of a node

Increasing the fault-tolerance threshold f may degrade *reliability*, if nodes are not *rejuvenated* and the mission time is large.

Another model

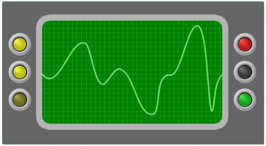
What if all nodes are compromised (e.g., leaky) from the start?

Another model

What if all nodes are compromised (e.g., leaky) from the start?

Threshold scheme may still be effective, if it increases the cost of exploitation!

(e.g., if exploiting a leakage vulnerability requires exponential number of traces for high-order Differential Power Analysis)



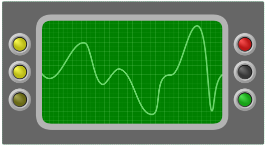
openclipart.org/detail/172330

Another model

What if all nodes are compromised (e.g., leaky) from the start?

Threshold scheme may still be effective,
if it increases the cost of exploitation!

(e.g., if exploiting a leakage vulnerability requires exponential number of traces for high-order Differential Power Analysis)



openclipart.org/detail/172330

Challenge questions:

- which models are realistic / match state-of-the-art attacks?
- what concrete parameters (e.g., n) thwart real attacks?

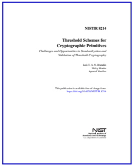
Outline 3

1. Crypto standards at NIST
2. Threshold intro
3. Threshold project
4. Threshold preliminary roadmap
5. Concluding remarks

NIST Internal Report (NISTIR) 8214

NIST Internal Report (NISTIR) 8214

Threshold Schemes for Cryptographic Primitives — Challenges and Opportunities in Standardization and Validation of Threshold Cryptography. (doi:10.6028/NIST.IR.8214)


NISTIR 8214

Threshold Schemes for Cryptographic Primitives

Challenges and Opportunities in Standardization and Validation of Threshold Cryptography

Luís T. A. N. Brandão
Nicky Mouha
Apostol Vassilev

This publication is available free of charge from:
https://doi.org/10.6028/NISTIR.8214

NIST
National Institute of Standards and Technology
U.S. Department of Commerce

https://csrc.nist.gov/publications/detail/nistir/8214/final

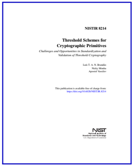
NIST Internal Report (NISTIR) 8214

Threshold Schemes for Cryptographic Primitives — Challenges and Opportunities in Standardization and Validation of Threshold Cryptography. (doi:10.6028/NIST.IR.8214)

The report sets a basis for discussion:

- need to characterize threshold schemes
- need to engage with stakeholders
- need to define criteria for standardization

Image adapted from:
openclipart.org/detail/283392



NISTIR 8214

Threshold Schemes for
Cryptographic Primitives

Challenges and Opportunities in Standardization and Validation of Threshold Cryptography

Luís T. A. N. Brandão
Nicky Mouha
Apostol Vassilev

This publication is available free of charge from:
https://doi.org/10.6028/NIST.IR.8214

NIST
National Institute of Standards and Technology
U.S. Department of Commerce

https://csrc.nist.gov/publications/detail/nistir/8214/final

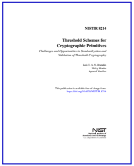
NIST Internal Report (NISTIR) 8214

Threshold Schemes for Cryptographic Primitives — Challenges and Opportunities in Standardization and Validation of Threshold Cryptography. (doi:10.6028/NIST.IR.8214)

The report sets a basis for discussion:

- need to characterize threshold schemes
- need to engage with stakeholders
- need to define criteria for standardization

Image adapted from: openclipart.org/detail/283392



NISTIR 8214

Threshold Schemes for Cryptographic Primitives

Challenges and Opportunities in Standardization and Validation of Threshold Cryptography

Luís T. A. N. Brandão
Nicky Mouha
Apostol Vassilev

This publication is available free of charge from:
https://doi.org/10.6028/NIST.IR.8214

Past timeline:

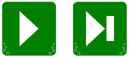
- 2018-July: Draft online 3 months for public comments
- 2018-October: Received comments from 13 external sources
- 2019-March: Final version online, along with "diff" and received comments

https://csrc.nist.gov/publications/detail/nistir/8214/final

Characterizing threshold schemes

Characterizing threshold schemes

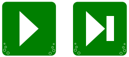
To reflect on a threshold scheme, start by characterizing **4 main features**:

- Kinds of threshold 
- Communication interfaces 
- Executing platform 
- Setup and maintenance 

The cliparts are from openclipart.org/detail/$*$, with $* \in \{71491, 190624, 101407, 161401, 161389\}$

Characterizing threshold schemes

To reflect on a threshold scheme, start by characterizing **4 main features**:

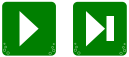
- Kinds of threshold 
- Executing platform 
- Communication interfaces 
- Setup and maintenance 

The cliparts are from openclipart.org/detail/$*$, with $* \in \{71491, 190624, 101407, 161401, 161389\}$

Each feature spans distinct options that affect security in different ways.

Characterizing threshold schemes

To reflect on a threshold scheme, start by characterizing **4 main features**:

- Kinds of threshold 
- Executing platform 
- Communication interfaces 
- Setup and maintenance 

The cliparts are from openclipart.org/detail/$*$, with $* \in \{71491, 190624, 101407, 161401, 161389\}$

Each feature spans distinct options that affect security in different ways.

A characterization provides a better context for security assertions.

Characterizing threshold schemes

To reflect on a threshold scheme, start by characterizing **4 main features**:

- Kinds of threshold
- Executing platform
- Communication interfaces
- Setup and maintenance

The cliparts are from openclipart.org/detail/$*$, with $* \in \{71491, 190624, 101407, 161401, 161389\}$

Each feature spans distinct options that affect security in different ways.

A characterization provides a better context for security assertions.

But there are other factors ...

Deployment context

Deployment context

- **Application context.** Should it affect security requirements?

Deployment context

- **Application context.** Should it affect security requirements?
  - signature correctness — may be deferred to client
  - decryption correctness — may require *robust* protocol

clker.com/clipart-3712.html

clker.com/clipart-encryption.html

Deployment context

- **Application context.** Should it affect security requirements?
  - signature correctness — may be deferred to client
  - decryption correctness — may require *robust* protocol

clker.com/ clipart-3712.html

clker.com/ clipart-encryption.html

- **Conceivable attack types.**

clker.com/clipart-10778

  - Active vs. passive
  - Static vs. adaptive
  - Stealth vs. detected
  - Invasive (physical) vs. non-invasive
  - Side-channel vs. communication interfaces
  - Parallel vs. sequential (wrt attacking nodes)

Deployment context

- **Application context.** Should it affect security requirements?
  - signature correctness — may be deferred to client
  - decryption correctness — may require *robust* protocol

clker.com/ clipart-3712.html

clker.com/ clipart-encryption.html

- **Conceivable attack types.**

clker.com/clipart-10778

  - Active vs. passive
  - Static vs. adaptive
  - Stealth vs. detected
  - Invasive (physical) vs. non-invasive
  - Side-channel vs. communication interfaces
  - Parallel vs. sequential (wrt attacking nodes)

A threshold scheme **improving** security against an attack in an application **may be powerless or degrade** security for another attack in another application

The validation challenge

The validation challenge

Devise standards of testable and validatable threshold schemes

vs.

devise testing and validation for standardized threshold schemes

The validation challenge

Devise standards of testable and validatable threshold schemes
vs.
devise testing and validation for standardized threshold schemes

Validation is needed in the federal context:

- need to use **validated** implementations [tC96] of **standardized** algorithms
- FIPS 140-2/3 defines, for cryptographic modules, 4 security levels: subsets of applicable security assertions [NIS01, NIS19]

(FIPS = Federal Information Processing Standards)

#NTCW2019

NIST Threshold Cryptography Workshop 2019

https://csrc.nist.gov/Events/2019/NTCW19

#NTCW2019

NIST Threshold Cryptography Workshop 2019

March 11–12, 2019 @
NIST Gaithersburg MD, USA

www.nist.gov/image/surfgaithersburg.jpg

https://csrc.nist.gov/Events/2019/NTCW19

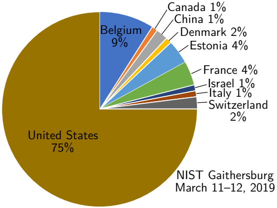
#NTCW2019

NIST Threshold Cryptography Workshop 2019

March 11–12, 2019 @
NIST Gaithersburg MD, USA

www.nist.gov/image/surfgaithersburg.jpg

Proportions of registrations
per country of affiliation



About 80 attendees

https://csrc.nist.gov/Events/2019/NTCW19

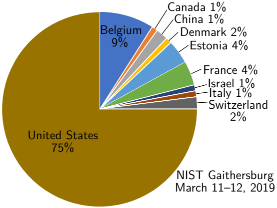
#NTCW2019

NIST Threshold Cryptography Workshop 2019

March 11–12, 2019 @
NIST Gaithersburg MD, USA

www.nist.gov/image/surfgaithersburg.jpg

Proportions of registrations
per country of affiliation



About 80 attendees

A platform for open interaction:

- hear about experiences with threshold crypto;
- get to know stakeholders;
- get input to reflect on roadmap and criteria.

https://csrc.nist.gov/Events/2019/NTCW19

Format and content

Format and content

Accepted 15 external submissions:

- 2 panels
- 5 papers
- 8 presentations

Format and content

Accepted 15 external submissions:

- 2 panels
- 5 papers
- 8 presentations

Plus:

- 2 invited keynotes
- 4 NIST talks
- 2 feedback moments

Format and content

Accepted 15 external submissions:

- 2 panels
- 5 papers
- 8 presentations

Plus:

- 2 invited keynotes
- 4 NIST talks
- 2 feedback moments

Videos, papers and presentations online at the NTCW webpage: https://csrc.nist.gov/Events/2019/NTCW19

Format and content

Accepted 15 external submissions:

- 2 panels
- 5 papers
- 8 presentations

Plus:

- 2 invited keynotes
- 4 NIST talks
- 2 feedback moments

Videos, papers and presentations online at the NTCW webpage: https://csrc.nist.gov/Events/2019/NTCW19

Discussion of diverse topics:

- threshold schemes in general (motivation and implementation feasibility);
- NIST standardization of cryptographic primitives
- a post-quantum threshold public-key encryption scheme;
- threshold signatures (adaptive security; elliptic curve digital signature algorithm);
- validation of cryptographic implementations;
- threshold circuit design (tradeoffs, pitfalls, combined attacks, verification tools);
- secret-sharing with leakage resilience;
- distributed symmetric-key encryption;
- applications and experience with threshold cryptography.

Results

Results

A step in *driving* *an open and transparent process* *towards standardization* of *threshold schemes for cryptographic primitives*. (See NISTIR 7977)

Results

A step in *driving* *an open and transparent process* *towards standardization of threshold schemes for cryptographic primitives*. (See NISTIR 7977)

Some notes:

- differences in granularity (building blocks vs. full functionalities);
- separation of single-device vs. multi-party;
- importance of envisioning applications;
- stakeholders' willingness to contribute;
- usefulness of explaining rationale (e.g., as complimented for the NISTIR);
- encouragement to move forward.

Results

A step in *driving <u>an open and transparent process</u> towards standardization of threshold schemes for cryptographic primitives*. (See NISTIR 7977)

Some notes:

- differences in granularity (building blocks vs. full functionalities);
- separation of single-device vs. multi-party;
- importance of envisioning applications;
- stakeholders' willingness to contribute;
- usefulness of explaining rationale (e.g., as complimented for the NISTIR);
- encouragement to move forward.

These elements are helpful for the next step ... designing a roadmap

Outline 4

1. Crypto standards at NIST

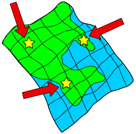
2. Threshold intro

3. Threshold project

4. Threshold preliminary roadmap

5. Concluding remarks

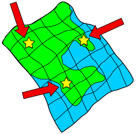
A “preliminary” roadmap



clker.com/clipart-15840.html

1. getting a map
2. deciding where to go
3. thinking how to get there

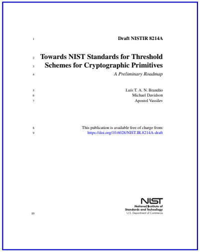
A “preliminary” roadmap



clker.com/clipart-15840.html

1. getting a map
2. deciding where to go
3. thinking how to get there

NISTIR 8214A (Draft), “Towards NIST Standards for Threshold Schemes for Cryptographic Primitives: A **Preliminary Roadmap** (doi:10.6028/NIST.IR.8214A-draft)



Draft NISTIR 8214A

Towards NIST Standards for Threshold Schemes for Cryptographic Primitives

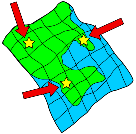
A Preliminary Roadmap

Luís T. A. N. Brandão
Michael Davidson
Apostol Vassilev

This publication is available free of charge from:
https://doi.org/10.6028/NIST.IR.8214A-draft

NIST
National Institute of Standards and Technology
U.S. Department of Commerce

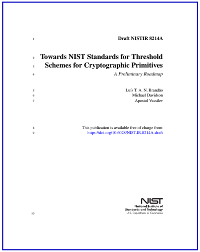
A "preliminary" roadmap



clker.com/clipart-15840.html

1. getting a map (**mapping layers**)
2. deciding where to go **(weighting factors)**
3. thinking how to get there **(collaboration)**

NISTIR 8214A (Draft), "Towards NIST Standards for Threshold Schemes for Cryptographic Primitives: A **Preliminary Roadmap** (doi:10.6028/NIST.IR.8214A-draft)



1 Draft NISTIR 8214A

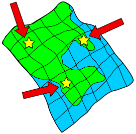
2 Towards NIST Standards for Threshold
3 Schemes for Cryptographic Primitives
4 *A Preliminary Roadmap*

5 Luís T. A. N. Brandão
6 Michael Davidson
7 Apostol Vassilev

8 This publication is available free of charge from:
9 https://doi.org/10.6028/NIST.IR.8214A-draft

10 NIST National Institute of Standards and Technology U.S. Department of Commerce

A "preliminary" roadmap



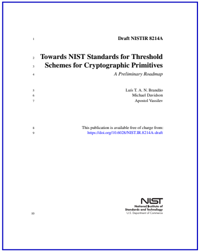
clker.com/clipart-15840.html

1. getting a map (**mapping layers**)
2. deciding where to go **(weighting factors)**
3. thinking how to get there **(collaboration)**

NISTIR 8214A (Draft), "Towards NIST Standards for Threshold Schemes for Cryptographic Primitives: A **Preliminary Roadmap** (doi:10.6028/NIST.IR.8214A-draft)

Lays the basis towards a roadmap:

- Map/organize potential items for standardization
- Motivating applications
- Features to consider
- Levels of difficulty / complexity
- Solicit preliminary input
- Identify phases of the standardization effort



Draft NISTIR 8214A

Towards NIST Standards for Threshold Schemes for Cryptographic Primitives

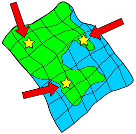
A Preliminary Roadmap

Luís T. A. N. Brandão
Michael Davidson
Apostol Vassilev

This publication is available free of charge from:
https://doi.org/10.6028/NIST.IR.8214A-draft

National Institute of Standards and Technology
U.S. Department of Commerce

A "preliminary" roadmap



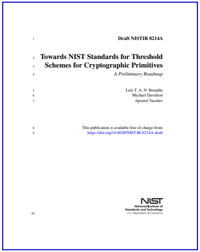
clker.com/clipart-15840.html

1. getting a map (**mapping layers**)
2. deciding where to go **(weighting factors)**
3. thinking how to get there **(collaboration)**

NISTIR 8214A (Draft), "Towards NIST Standards for Threshold Schemes for Cryptographic Primitives: A **Preliminary Roadmap** (doi:10.6028/NIST.IR.8214A-draft)

Lays the basis towards a roadmap:

- Map/organize potential items for standardization
- Motivating applications
- Features to consider
- Levels of difficulty / complexity
- Solicit preliminary input
- Identify phases of the standardization effort



Draft NISTIR 8214A

Towards NIST Standards for Threshold Schemes for Cryptographic Primitives

A Preliminary Roadmap

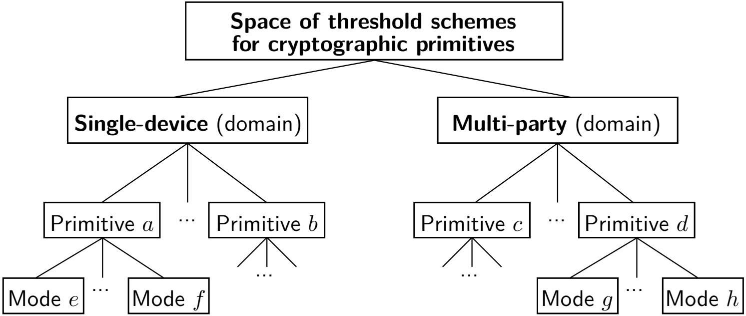
Luís T. A. N. Brandão
Michael Davidson
Apostol Vassilev

This publication is available free of charge from:
https://doi.org/10.6028/NIST.IR.8214A-draft

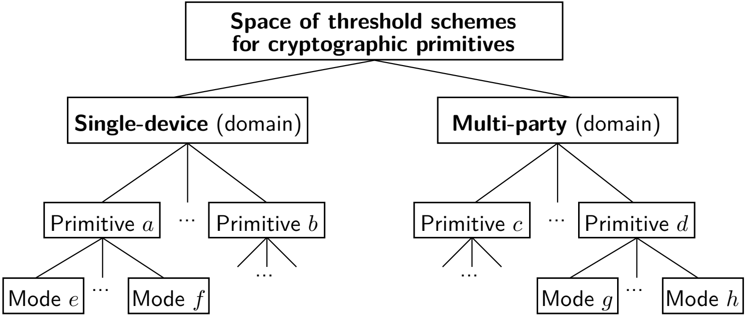
NIST
National Institute of Standards and Technology
U.S. Department of Commerce

Open to public comments: 2019/Nov/11 – 2020/Feb/10.

Mapping the space of potential “schemes”



Mapping the space of potential “schemes”



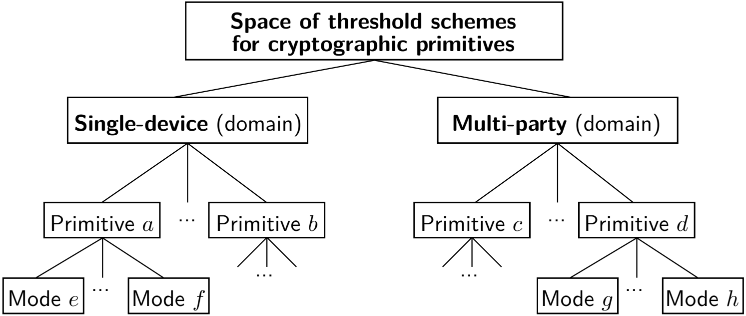
Single-device:

- rigid configuration of components
- strictly defined physical boundaries
- dedicated communication network

Multi-party:

- enable modularized patching of components
- possible dynamic configurations of parties
- some distributed systems’ problems

Mapping the space of potential "schemes"



Single-device:

- rigid configuration of components
- strictly defined physical boundaries
- dedicated communication network

Multi-party:

- enable modularized patching of components
- possible dynamic configurations of parties
- some distributed systems' problems

Each ***domain*** also represents a ***track*** in the standardization effort.

Some conceivable primitives (focus on NIST-approved)

Less complex:

- Multi-party: RSA decrypt & sign; EdDSA/Schnorr* sign; ECC key-gen.
- Single-device: AES threshold circuit design against leakage.

Some conceivable primitives (focus on NIST-approved)

Less complex:

- Multi-party: RSA decrypt & sign; EdDSA/Schnorr* sign; ECC key-gen.
- Single-device: AES threshold circuit design against leakage.

More complex:

- Multi-party: ECDSA signature; RSA key-gen; AES enciphering.
- Single-device: AES threshold circuit against combined attacks.

Some conceivable primitives (focus on NIST-approved)

Less complex:

- Multi-party: RSA decrypt & sign; EdDSA/Schnorr* sign; ECC key-gen.
- Single-device: AES threshold circuit design against leakage.

More complex:

- Multi-party: ECDSA signature; RSA key-gen; AES enciphering.
- Single-device: AES threshold circuit against combined attacks.

Research interest (but not focus of standardization):

- Multi-party: post-quantum signing & PKE decryption; ...
- Single-device: threshold lightweight-crypto; ...

Some conceivable primitives (focus on NIST-approved)

Less complex:

- Multi-party: RSA decrypt & sign; EdDSA/Schnorr* sign; ECC key-gen.
- Single-device: AES threshold circuit design against leakage.

More complex:

- Multi-party: ECDSA signature; RSA key-gen; AES enciphering.
- Single-device: AES threshold circuit against combined attacks.

Research interest (but not focus of standardization):

- Multi-party: post-quantum signing & PKE decryption; ...
- Single-device: threshold lightweight-crypto; ...

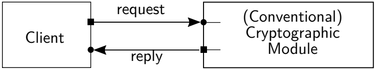
Notes:

- **Complexity:** depends on more factors, e.g., $*$; *mode* (next slide).
- **Other cases:** distributed RNG; some can have similarities across tracks.

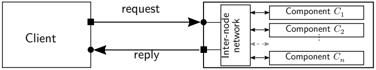
Threshold modes (features in the perspective of the client)

Threshold modes (features in the perspective of the client)

Input/Output interface: client communication with the module / threshold entity?



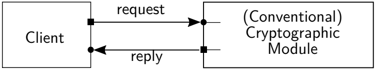
Conventional (non-threshold)



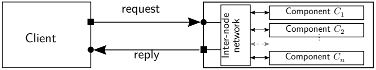
Not-shared-IO

Threshold modes (features in the perspective of the client)

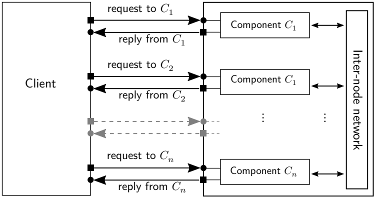
Input/Output interface: client communication with the module / threshold entity?



Conventional (non-threshold)



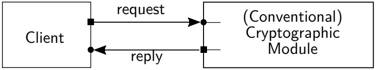
Not-shared-IO



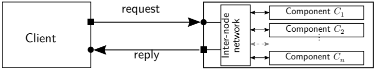
Shared-IO

Threshold modes (features in the perspective of the client)

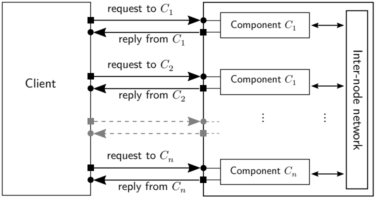
Input/Output interface: client communication with the module / threshold entity?



Conventional (non-threshold)



Not-shared-IO

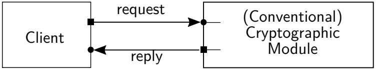


Shared-IO

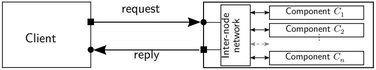
(Shared-I and Shared-O are other modes where only the **i**nput and only the **o**utput are shared, respectively)

Threshold modes (features in the perspective of the client)

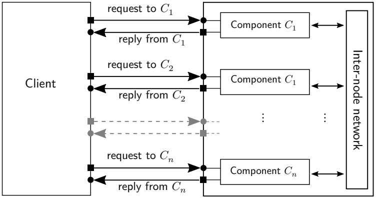
Input/Output interface: client communication with the module / threshold entity?



Conventional (non-threshold)



Not-shared-IO



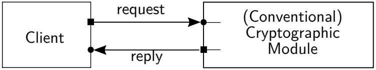
Shared-IO

(Shared-I and Shared-O are other modes where only the **i**nput and only the **o**utput are shared, respectively)

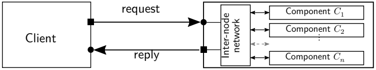
Auditability: can the client prove (or be convinced) the operation was thresholdized?

Threshold modes (features in the perspective of the client)

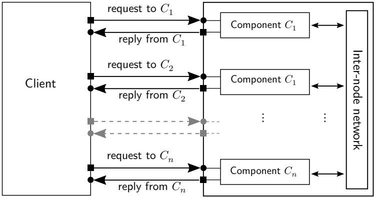
Input/Output interface: client communication with the module / threshold entity?



Conventional (non-threshold)



Not-shared-IO



Shared-IO

(Shared-I and Shared-O are other modes where only the **i**nput and only the **o**utput are shared, respectively)

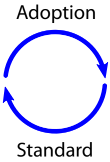
Auditability: can the client prove (or be convinced) the operation was thresholdized?

Examples:

- Shared-I: signature protecting the secrecy of the input
- Shared-O: decryption protecting the secrecy of the output
- Auditable: succinct multi-signature verifiable against several public-keys

Standardization vs. adoption

"not every conceivable possibility is suitable for standardization"

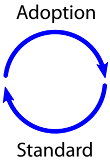


Standardization vs. adoption

"not every conceivable possibility is suitable for standardization"

Example motivating applications:

1. Secrets protected at rest (e.g., for high-value signature keys)
2. Confidential communication (e.g., via shared-O decryption)
3. Distributed key generation (e.g., to avoid dealers)
4. Leakage-resistant hardware (e.g., via threshold circuit design)
5. Accountable transactions (e.g., via multi-signatures)
6. Password authentication (e.g., via threshold hashing)
7. Distributed computation (e.g., across HSMs or VMs)



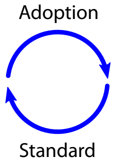
Standardization vs. adoption

"not every conceivable possibility is suitable for standardization"

Example motivating applications:

1. Secrets protected at rest (e.g., for high-value signature keys)
2. Confidential communication (e.g., via shared-O decryption)
3. Distributed key generation (e.g., to avoid dealers)
4. Leakage-resistant hardware (e.g., via threshold circuit design)
5. Accountable transactions (e.g., via multi-signatures)
6. Password authentication (e.g., via threshold hashing)
7. Distributed computation (e.g., across HSMs or VMs)

We find useful to hear stakeholders' insights, to "focus on where there is a high need and high potential for adoption" ...



Standardization vs. adoption

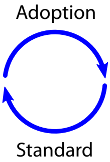
“not every conceivable possibility is suitable for standardization”

Example motivating applications:

1. Secrets protected at rest (e.g., for high-value signature keys)
2. Confidential communication (e.g., via shared-O decryption)
3. Distributed key generation (e.g., to avoid dealers)
4. Leakage-resistant hardware (e.g., via threshold circuit design)
5. Accountable transactions (e.g., via multi-signatures)
6. Password authentication (e.g., via threshold hashing)
7. Distributed computation (e.g., across HSMs or VMs)

We find useful to hear stakeholders’ insights, to “focus on where there is a high need and high potential for adoption” ...

best practices; *minimum defaults*; *interoperability*; *innovation*.



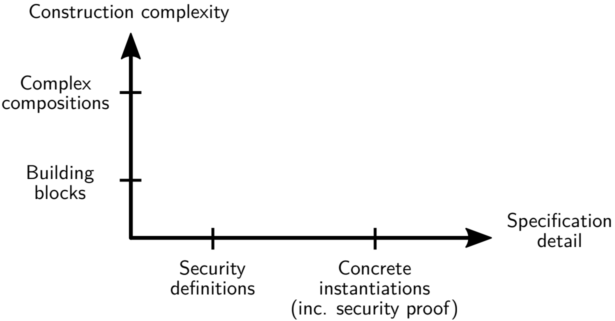
The modularity challenge

Do we need to compromise between:

The modularity challenge

Do we need to compromise between:

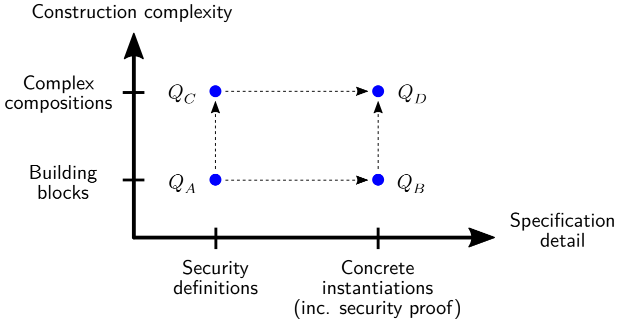
- ideal functionalities vs. concrete protocols of threshold schemes?
- building blocks vs. complex constructions?



The modularity challenge

Do we need to compromise between:

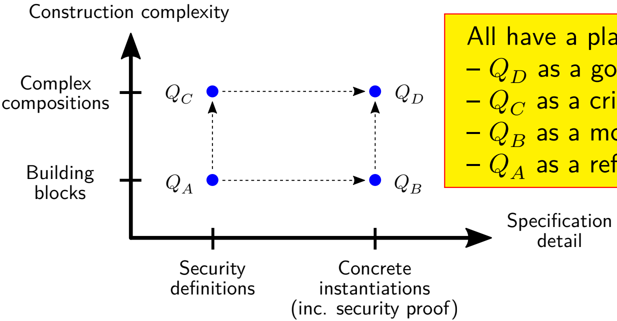
- ideal functionalities vs. concrete protocols of threshold schemes?
- building blocks vs. complex constructions?



The modularity challenge

Do we need to compromise between:

- ideal functionalities vs. concrete protocols of threshold schemes?
- building blocks vs. complex constructions?

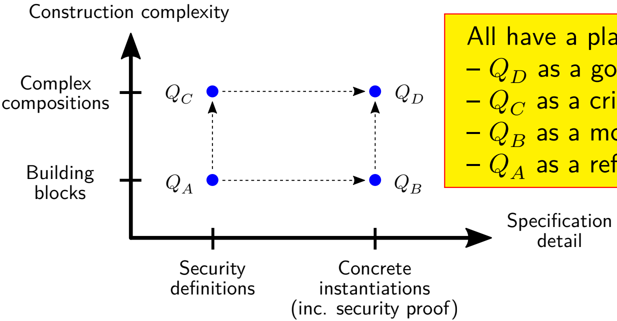


All have a place in the process:
– Q_D as a goal;
– Q_C as a criterion;
– Q_B as a module;
– Q_A as a reference definition.

The modularity challenge

Do we need to compromise between:

- ideal functionalities vs. concrete protocols of threshold schemes?
- building blocks vs. complex constructions?



All have a place in the process:
– Q_D as a goal;
– Q_C as a criterion;
– Q_B as a module;
– Q_A as a reference definition.

Example possible *gadgets*: secret sharing; distributed/correlated randomness; consensus; oblivious transfer; garbled circuits; ...

Designing concrete threshold schemes

Additional features to consider:

- Configurability of threshold parameters
- Rejuvenation of components (shares, parties, ...)
- Security (functionality/properties): composable?, adaptive?, graceful degradation?, ...
- Suitability for testing and validation
- ...

Designing concrete threshold schemes

Additional features to consider:

- Configurability of threshold parameters
- Rejuvenation of components (shares, parties, ...)
- Security (functionality/properties): composable?, adaptive?, graceful degradation?, ...
- Suitability for testing and validation
- ...

Important:

- Useful to get feedback from stakeholders about concrete examples
- These may held define criteria for calls / evaluation / selection

Development process

Generic possible sequence of phases:

1. Roadmap → 2. Calls with criteria → 3. Evaluation → 4. Issue standards

(Each phase to include public feedback. Some Threshold Cryptography workshops along the way?)

Development process

Generic possible sequence of phases:

1. Roadmap → 2. Calls with criteria → 3. Evaluation → 4. Issue standards

(Each phase to include public feedback. Some Threshold Cryptography workshops along the way?)

Different standardization *items* can have **different**:

- **calls for contributions:** feedback on reference protocols; new protocols; reference implementations showing feasibility; research results, ...
- **timelines** (e.g., depending on complexity; existing rationale for choices)
- **final formats:** addendum vs. standalone standard, reference to other standards, implementation/validation guidelines, reference definitions,

Public feedback: a main pillar of the process

Promotes: openness, transparency and scrutiny, technical merit, trust, ...

Public feedback: a main pillar of the process

Promotes: openness, transparency and scrutiny, technical merit, trust, ...

Useful feedback now — potential to shape the roadmap and criteria:

- **Standardization *items*:** domain / primitive / mode;
- **Context:** application motivation, deployment setting, adversarial model
- **Desirable features:** rejuvenation, dynamic thresholds; robustness; composability; testability; ...
- **Concrete protocols/algorithms:** comparisons of state-of-the-art references
- **Reference implementations:** feasibility, benchmarks, open source, ...
- **Intellectual property:** information on known patents, licenses, ...

Public feedback: a main pillar of the process

Promotes: openness, transparency and scrutiny, technical merit, trust, ...

Useful feedback now — potential to shape the roadmap and criteria:

- **Standardization *items*:** domain / primitive / mode;
- **Context:** application motivation, deployment setting, adversarial model
- **Desirable features:** rejuvenation, dynamic thresholds; robustness; composability; testability; ...
- **Concrete protocols/algorithms:** comparisons of state-of-the-art references
- **Reference implementations:** feasibility, benchmarks, open source, ...
- **Intellectual property:** information on known patents, licenses, ...

Useful feedback later:

- Answers to subsequent calls for contributions

Intellectual property claims

The topic of intellectual property is relevant:

Intellectual property claims

The topic of intellectual property is relevant:

- Asking for disclosure of patents: *call* for disclosure, conditions for submitting
- Promote "FRAND" license: **f**air, **r**easonable, **a**nd **n**on-**d**iscriminatory*

  * the NIST-ITL patent policy puts it as "reasonable and demonstrably free from unfair discrimination"

- Cannot force third party to disclose or enable FRAND terms ... but can choose to specify guidance based on expectation of FRAND terms.

Intellectual property claims

The topic of intellectual property is relevant:

- Asking for disclosure of patents: *call* for disclosure, conditions for submitting
- Promote "FRAND" license: **f**air, **r**easonable, **a**nd **n**on-**d**iscriminatory*

  * the NIST-ITL patent policy puts it as "reasonable and demonstrably free from unfair discrimination"
- Cannot force third party to disclose or enable FRAND terms ... but can choose to specify guidance based on expectation of FRAND terms.

Excerpt from NIST-ITL patent policy: "*assurance [...] that [...] party does not hold [...] any essential patent claim(s); or that a license [...] will be made available [...] under reasonable terms and conditions that are demonstrably free of any unfair discrimination;*" [possibly without compensation]

Excerpt from NISTIR 7977: "*NIST has noted a strong preference among its users for solutions that are unencumbered by royalty-bearing patented technologies. NIST has observed that widespread adoption of cryptographic solutions that it has developed has been facilitated by royalty-free licensing terms.*" [...]

"*NIST will explicitly* recognize and respect the value of IP *and the need to protect IP if it is incorporated into standards or guidelines.*"

Outline 5

1. Crypto standards at NIST
2. Threshold intro
3. Threshold project
4. Threshold preliminary roadmap
5. Concluding remarks

Concluding remarks

Concluding remarks

- NIST-CSD is driving an effort to standardize threshold schemes for NIST-approved cryptographic primitives
- Collaboration with stake-holders is essential
- We are in the stage of building a roadmap ... your feedback can (and should) help determine the outcome
- A two track approach (multi-party and single-device)
- Various standardization items in each track, with various complexities

The test of time

70 years from now, will *threshold schemes* (still) be used to enable distributed trust in the implementation and operation of cryptographic primitives?

The test of time

70 years from now, will *threshold schemes* (still) be used to enable distributed trust in the implementation and operation of cryptographic primitives?

Photo in 1948 *

Photo in 2018: https://www.nist.gov/sites/default/files/documents/2018/06/15/nist_gaithersburg_master_plan_may_7_2018.pdf

The NIST Stone Test Wall: "Constructed [in 1948] to study the performance of stone subjected to weathering. It contains 2352 individual samples of stone, of which 2032 are domestic stone from 47 states, and 320 are stones from 16 foreign countries."

* https://www.nist.gov/el/materials-and-structural-systems-division-73100/nist-stone-wall

- Project webpage: https://csrc.nist.gov/Projects/Threshold-Cryptography
- Project email adress: threshold-crypto@nist.gov
- NISTIR 8214: https://csrc.nist.gov/publications/detail/nistir/8214/final
- NISTIR 8214A (draft): https://csrc.nist.gov/publications/detail/nistir/8214a/draft
- TC-forum: https://list.nist.gov/tc-forum

- Project webpage: https://csrc.nist.gov/Projects/Threshold-Cryptography
- Project email adress: threshold-crypto@nist.gov
- NISTIR 8214: https://csrc.nist.gov/publications/detail/nistir/8214/final
- NISTIR 8214A (draft): https://csrc.nist.gov/publications/detail/nistir/8214a/draft
- TC-forum: https://list.nist.gov/tc-forum

Word cloud based on the NISTIR 8214

Thank you for your attention

- Project webpage: https://csrc.nist.gov/Projects/Threshold-Cryptography
- Project email adress: threshold-crypto@nist.gov
- NISTIR 8214: https://csrc.nist.gov/publications/detail/nistir/8214/final
- NISTIR 8214A (draft): https://csrc.nist.gov/publications/detail/nistir/8214a/draft
- TC-forum: https://list.nist.gov/tc-forum

Word cloud based on the NISTIR 8214

Presentation at the Theory of Implementation Security (TIS'19) Workshop
November 11, 2019 @ London, UK
luis.brandao@nist.gov

Disclaimer. Opinions expressed in this presentation are from the author(s) and are not to be construed as official or as views of the U.S. Department of Commerce. The identification of any commercial product or trade names in this presentation does not imply endorsement of recommendation by NIST, nor is it intended to imply that the material or equipment identified are necessarily the best available for the purpose.

Disclaimer. Some external-source images and cliparts were included/adapted in this presentation with the expectation of such use constituting licensed and/or fair use.

References

[BB12] L. T. A. N. Brandão and A. N. Bessani. *On the reliability and availability of replicated and rejuvenating systems under stealth attacks and intrusions*. Journal of the Brazilian Computer Society, 18(1):61–80, 2012. DOI:10.1007/s13173-012-0062-x.

[BDL97] D. Boneh, R. A. DeMillo, and R. J. Lipton. *On the Importance of Checking Cryptographic Protocols for Faults*. In W. Fumy (ed.), Advances in Cryptology — EUROCRYPT '97, pages 37–51, Berlin, Heidelberg, 1997. Springer Berlin Heidelberg. DOI:10.1007/3-540-69053-0_4.

[BDV19] L. T. A. N. Brandão, M. Davidson, and A. Vassilev. *Towards NIST Standards for Threshold Schemes for Cryptographic Primitives: A Preliminary Roadmap*. Draft NISTIR 8214A, November 2019. DOI:10.6028/NIST.IR.8214A-draft.

[BMV19] L. T. A. N. Brandão, N. Mouha, and A. Vassilev. *Threshold Schemes for Cryptographic Primitives: Challenges and Opportunities in Standardization and Validation of Threshold Cryptography*. NISTIR 8214, March 2019. DOI:10.6028/NIST.IR.8214.

[BMW+18] J. v. Bulck, M. Minkin, O. Weisse, D. Genkin, B. Kasikci, F. Piessens, M. Silberstein, T. F. Wenisch, Y. Yarom, and R. Strackx. *Foreshadow: Extracting the Keys to the Intel SGX Kingdom with Transient Out-of-Order Execution*. In 27th USENIX Security Symposium (USENIX Security 18), page 991–1008, Baltimore, MD, 2018. USENIX Association.

[DLK+14] Z. Durumeric, F. Li, J. Kasten, J. Amann, J. Beekman, M. Payer, N. Weaver, D. Adrian, V. Paxson, M. Bailey, and J. A. Halderman. *The Matter of Heartbleed*. In Proceedings of the 2014 Conference on Internet Measurement Conference, IMC '14, pages 475–488, New York, NY, USA, 2014. ACM. DOI:10.1145/2663716.2663755.

[Don13] D. Donzai. *Using Cold Boot Attacks and Other Forensic Techniques in Penetration Tests*, 2013. https://www.ethicalhacker.net/features/root/using-cold-boot-attacks-forensic-techniques-penetration-tests/. Accessed: July 2018.

[Gro16] C. T. Group. *NIST Cryptographic Standards and Guidelines Development Process*. NISTIR 7977, March 2016. DOI:10.6028/NIST.IR.7977.

[HSH+09] J. A. Halderman, S. D. Schoen, N. Heninger, W. Clarkson, W. Paul, J. A. Calandrino, A. J. Feldman, J. Appelbaum, and E. W. Felten. *Lest We Remember: Cold-boot Attacks on Encryption Keys*. Commun. ACM, 52(5):91–98, May 2009. DOI:10.1145/1506409.1506429.

[KGG+18] P. Kocher, D. Genkin, D. Gruss, W. Haas, M. Hamburg, M. Lipp, S. Mangard, T. Prescher, M. Schwarz, and Y. Yarom. *Spectre Attacks: Exploiting Speculative Execution*. ArXiv e-prints, January 2018. arXiv:1801.01203.

[LSG+18] M. Lipp, M. Schwarz, D. Gruss, T. Prescher, W. Haas, S. Mangard, P. Kocher, D. Genkin, Y. Yarom, and M. Hamburg. *Meltdown*. ArXiv e-prints, jan 2018. arXiv:1801.01207.

[MDS19] MDS. *RIDL and Fallout: MDS attacks*, 2019. https://mdsattacks.com/.

[NIS01] NIST. *Security Requirements for Cryptographic Modules, Federal Information Processing Standard (FIPS) 140-2*, 2001. DOI:10.6028/NIST.FIPS.140-2.

[NIS19] NIST. *Security Requirements for Cryptographic Modules, Federal Information Processing Standard (FIPS) 140-3*, 2019. DOI:10.6028/NIST.FIPS.140-3.

[RSA78] R. L. Rivest, A. Shamir, and L. Adleman. *A method for obtaining digital signatures and public-key cryptosystems*. Communications of the ACM, 21(2):120–126, 1978. DOI:10.1145/359340.359342.

[RSWO17] E. Ronen., A. Shamir, A.-O. Weingarten, and C. O'Flynn. *IoT Goes Nuclear: Creating a ZigBee Chain Reaction*. IEEE Symposium on Security and Privacy, pages 195–212, 2017. DOI:10.1109/SP.2017.14.

[SH07] J.-M. Schmidt and M. Hutter. *Optical and EM Fault-Attacks on CRT-based RSA: Concrete Results*, pages 61–67. Verlag der Technischen Universität Graz, 2007.

[Sha79] A. Shamir. *How to Share a Secret*. Communications of the ACM, 22(11):612–613, Nov 1979. DOI:10.1145/359168.359176.

[tC96] U. S. 104th Congress. *Information Technology Management Reform Act. Public Law 104–106, Section 5131*, 1996. https://www.dol.gov/ocfo/media/regs/ITMRA.pdf.

[WBM+18] O. Weisse, J. v. Bulck, M. Minkin, D. Genkin, B. Kasikci, F. Piessens, M. Silberstein, R. Strackx, T. F. Wenisch, and Y. Yarom. *Foreshadow-NG: Breaking the Virtual Memory Abstraction with Transient Out-of-Order Execution*. Technical Report, 2018.

List of slides

1 Towards Standardization of Threshold ...
2 Outline
3 Outline 1
4 Some NIST data
5 Laboratories, divisions, groups
6 Some projects of crypto primitives
7 Some standardized cryptographic primitives
8 Other processes (examples)
9 Outline 2
10 Beyond defining basic crypto primitives?
11 Crypto can be affected by vulnerabilities!
12 The threshold approach
13 Secret Sharing Schemes (a starting point)
14 A simple example: RSA signature
15 What do the thresholds k and f mean?
15 Reliability $(\mathcal{R})$ as one metric of security
16 Another model
17 Outline 3
18 NIST Internal Report (NISTIR) 8214
19 Characterizing threshold schemes
20 Deployment context
21 The validation challenge
22 #NTCW2019
23 Format and content
24 Results
25 Outline 4
26 A "preliminary" roadmap
27 Mapping the space of potential "schemes"
28 Some conceivable primitives
29 Threshold modes
30 Standardization vs. adoption
31 The modularity challenge
32 Designing concrete threshold schemes
33 Development process
34 Public feedback: a main pillar of the process
35 Intellectual property claims
36 Outline 5
37 Concluding remarks
38 The test of time
39 **Thank you** for your attention
40 References
41 List of slides