

1

**Draft** NISTIR 8214

2

# Threshold Schemes for Cryptographic Primitives

3

4

*Challenges and Opportunities in Standardization and  
Validation of Threshold Cryptography*

5

6

Luís T. A. N. Brandão

7

Nicky Mouha

8

Apostol Vassilev

9

10

11

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

12



13

**Draft** NISTIR 8214

14

# Threshold Schemes for Cryptographic Primitives

15

16

*Challenges and Opportunities in Standardization and  
Validation of Threshold Cryptography*

17

18

Luís T. A. N. Brandão

19

Nicky Mouha

20

Apostol Vassilev

21

*Computer Security Division*

22

*Information Technology Laboratory*

23

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

24

~~July 2018~~ March 2019

25



26

U.S. Department of Commerce

27

*Wilbur L. Ross, Jr., Secretary*

28

National Institute of Standards and Technology

29

*Walter G. Copan, NIST Director and Under Secretary of Commerce for Standards and Technology*

30 National Institute of Standards and Technology Internal Report 8214  
31 ~~55 pages (July 2018)~~ 63 pages (March 2019)

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

33 Certain commercial entities, equipment, or materials may be identified in this document in order to describe an  
34 experimental procedure or concept adequately. Such identification is not intended to imply recommendation or  
35 endorsement by NIST, nor is it intended to imply that the entities, materials, or equipment are necessarily the  
36 best available for the purpose.

37 There may be references in this publication to other publications currently under development by NIST  
38 in accordance with its assigned statutory responsibilities. The information in this publication, including  
39 concepts and methodologies, may be used by federal agencies even before the completion of such companion  
40 publications. Thus, until each publication is completed, current requirements, guidelines, and procedures,  
41 where they exist, remain operative. For planning and transition purposes, federal agencies may wish to closely  
42 follow the development of these new publications by NIST.

43 Organizations are encouraged to review all draft publications during public comment periods and provide  
44 feedback to NIST. Many NIST cybersecurity publications, other than the ones noted above, are available at  
45 <https://csrc.nist.gov/publications>.

46 **Public comment period** Comments on this publication may be submitted  
47 to: *July 26, 2018, through October 22, 2018*

48 National Institute of Standards and Technology  
49 Attn: Computer Security Division, Information Technology Laboratory  
50 100 Bureau Drive (Mail Stop 8930) Gaithersburg, MD 20899-8930  
51 Email: [threshold-crypto@nist.gov](mailto:threshold-crypto@nist.gov)

52 All comments are subject to release under the Freedom of Information Act (FOIA).

53 **Reports on Computer Systems Technology**

54 The Information Technology Laboratory (ITL) at the National Institute of Standards and  
55 Technology (NIST) promotes the U.S. economy and public welfare by providing technical  
56 leadership for the Nation’s measurement and standards infrastructure. ITL develops tests,  
57 test methods, reference data, proof of concept implementations, and technical analyses to  
58 advance the development and productive use of information technology. ITL’s responsi-  
59 bilities include the development of management, administrative, technical, and physical  
60 standards and guidelines for the cost-effective security and privacy of other than national  
61 security-related information in federal information systems.

62 **Abstract**

63 The Computer Security Division at the National Institute of Standards and Technology is  
64 interested in promoting the security of implementations of cryptographic primitives. This se-  
65 curity depends not only on the theoretical properties of the primitives but also on the ability to  
66 withstand attacks on their implementations. It is thus important to mitigate breakdowns that  
67 result from differences between ideal and real implementations of cryptographic algorithms.

68 This document overviews ~~threshold cryptographic schemes, which enable attaining~~  
69 the possibility of implementing cryptographic primitives using threshold schemes, where R1: NI  
70 multiple components contribute to the operation in a way that attains the desired security  
71 goals even if  $f$  out of  $n$  of its components are compromised. There is also an identified poten-  
72 tial in providing resistance against side-channel attacks, which exploit inadvertent leakage  
73 from real implementations. Security goals of interest include the secrecy of cryptographic  
74 keys, as well as enhanced integrity and availability, among others.

75 This document considers challenges and opportunities related to standardization of  
76 threshold schemes for cryptographic primitives. It includes examples illustrating security  
77 tradeoffs under variations of system model and adversaries. It enumerates several high-level  
78 characterizing features of threshold schemes, including the types of threshold, the communi-  
79 cation interfaces (with the environment and between components), the executing platform  
80 (e.g., single device vs. multiple devices) and the setup and maintenance requirements.

81 The document poses a number of questions, motivating aspects to take into account when  
82 considering standardization. A particular challenge is the development of criteria that may  
83 help guide a selection of threshold cryptographic schemes. An open question is deciding at  
84 what level each standard should be defined (e.g., specific base techniques vs. conceptualized  
85 functionalities) and which flexibility of parametrization they should allow. Suitability to  
86 testing and validation of implementations are also major concerns to be addressed. Overall,  
87 the document intends to support discussion about standardization, including motivating  
88 an engagement from stakeholders. This is a step towards enabling threshold cryptography  
89 within the US federal government and beyond.

90 **Keywords:** threshold schemes; secure implementations; cryptographic primitives; thresh-  
91 old cryptography; secure multi-party computation; intrusion tolerance; distributed systems;

92 resistance to side-channel attacks; standards and validation.

93 **Acknowledgments**

94 The authors thank their [NIST](#) colleagues who reviewed recent or early versions of this ~~draft~~  
95 ~~publication document~~. This includes Lily Chen, René Peralta, Ray Perlner, and Andrew  
96 Regenscheid. ~~We look forward to receiving further feedback during the phase of public~~  
97 ~~comments.~~

98 The initial draft of this report was published online on July 26, 2018, for a period of  
99 public comments. We have since then received valuable feedback from external reviewers,  
100 which allowed us to improve the quality of the final document. We thank the comments  
101 received (in chronological order) from the following persons: Svetla Nikova from KU  
102 Leuven; Gokhan Kocak from Asena Inc.; Oliver Stengele from the Karlsruhe Institute  
103 of Technology; Aivo Kalu from Cybernetica AS; Christian Cachin, Hugo Krawczyk, Tal  
104 Rabin, Jason Resch and Chrysa Stathakopoulou from IBM Research; Tanja Lange from  
105 Technische Universiteit Eindhoven; Yehuda Lindel from Bar-Ilan University and Unbound  
106 Tech; Samuel Ranellucci from Unbound Tech; Dan Bogdanov from Cybernetica AS;  
107 Rosario Gennaro from The City College of New York; Thalia May Laing from HP Labs;  
108 Karim Eldefrawy from SRI International; John Wallrabenstein from Analog.

R2: N2

## 109 **Executive Summary**

110 As cryptography becomes ubiquitous, it becomes increasingly relevant to address the  
111 potentially disastrous breakdowns resulting from differences between ideal and real imple-  
112 mentations of cryptographic algorithms. These differences give rise to a range of attacks that  
113 exploit vulnerabilities in order to compromise diverse aspects of real-world implementations.  
114 Threshold schemes have the potential to enable secure modes of operation even when certain  
115 subsets of components are compromised. However, they also present new challenges for  
116 the standardization and validation of security assertions about their implementations.

117 This report is focused on threshold cryptographic schemes, i.e., threshold schemes  
118 used for secure implementations of cryptographic primitives. In an ~~*f-out-of-n*~~ a threshold  
119 scheme, some security property is tolerant to the compromise of up to a threshold number  
120 ~~*f out-of*~~ (out of a total number *n* of) components in the system. The topic is related to  
121 traditional “threshold cryptography” (here adopted as an umbrella term), secure multi-party  
122 computation and intrusion-tolerant distributed systems. A major goal is enhanced protection  
123 of secret keys used by implementations of cryptographic algorithms. More generally, the  
124 goal includes the enhancement of a variety of security properties, such as confidentiality,  
125 integrity and/or availability.

R3: A2, E45,  
E60, F2, K11,  
K12

126 Secret sharing is a fundamental technique in threshold cryptography. It enables a key (or  
127 some other secret input) to be split into multiple shares distributed across multiple parties.  
128 The “threshold” property translates into the ability to reconstruct the key from a threshold  
129 number of shares, but not from fewer. Thus, splitting a key into shares is an approach for  
130 protecting the secrecy of a key at rest, since the leakage of one or few shares does not reveal  
131 the key. However, this does not solve the problem of how to execute an algorithm that  
132 depends on a key. Particularly, conventional implementations of key-based cryptographic  
133 algorithms require the whole key as input, so if the key had been subject to secret sharing  
134 then the shared key would have to be reconstructed for use by the algorithm.

135 In threshold cryptography, the shares of the key do not need to be recombined to compute  
136 a particular result. Instead, the parties independently or collaboratively calculate shares  
137 of the output, without revealing the input shares to one another. This may be facilitated  
138 by certain mathematical properties, such as homomorphisms, or by cryptographic “secure  
139 computation” protocols. Using the threshold property, the output from the share computation  
140 can then be reconstructed into a final output. This is possible to achieve for NIST-approved  
141 algorithms, such as ~~*RSA and DSA*~~ Rivest–Shamir–Adleman (RSA) and Digital Signature  
142 Algorithm (DSA) signatures, and ~~*AES*~~ Advanced Encryption Standard (AES) enciphering  
143 and deciphering.

R4: N3

144 Threshold schemes can be used, with different security goals, in different applications.  
145 For example: (i) implement a digital signature algorithm without any single component  
146 ever holding the signing key; (ii) implement encryption and decryption correctly even if one  
147 compromised component attempts to corrupt the output; (iii) generate unbiased randomness

148 even if some (but not all) randomness contributors are biased or unavailable.

R5: A1

149 The computational paradigm in threshold cryptography brings several security advan-  
150 tages but also some potential weaknesses. For example, the use of multiple shares increases  
151 the attack surface to encompass all shares. Thus, the security effect of implementing a  
152 threshold scheme depends on an attack model. It is particularly relevant to consider how  
153 difficult ~~may be the compromise of~~ it may be to compromise more than the threshold num-  
154 ber  $f$  of components. In some cases, for example with low  $f$ , the increased attack surface  
155 may enable an attack more efficient and effective than possible against a conventional  
156 (non-threshold) primitive. ~~—, even if the nodes in the threshold scheme have independent~~  
157 modes of compromise (e.g., each compromisable via mutually exclusive attack vectors).  
158 On the other hand, a threshold scheme may provide better security even if the components  
159 are individually easier to compromise, e.g., in some settings/models where they are also  
160 easier to patch.

R6: N9

R7: A7, G4,  
E14, E16

161 The security effect of a threshold design may also be different across different properties  
162 of interest. For example, while the compromise of one share might not reveal the original key,  
163 the corruption of a single share (or of a computation dependent on it) may affect the integrity  
164 of the output. These observations highlight the need to look at the security benefits brought  
165 by each threshold scheme as a possible tradeoff across properties. In some settings there may  
166 be a strengthening of some security properties while for others the assurance may be reduced.

167 There are techniques designed to mitigate foreseen compromises in more complicated  
168 scenarios. For example, verifiable secret-sharing enables detection of misuse of shares by a  
169 shareholder, thereby enabling operational modes that tolerate this kind of corruption. As an-  
170 other example, proactive secret sharing can be used to periodically reshare a secret, thereby  
171 periodically reducing to zero the number of compromised shares. Assuming that old un-  
172 compromised shares are erased, the refreshing makes it more difficult to reach a state where  
173 the number of contemporaneous compromised shares surpasses the compromise threshold.

174 Separating the analysis of different security aspects can sometimes lead to pitfalls. To  
175 avoid such problems it is important to use appropriate formal models of security. At the  
176 same time, it is relevant to assess potential tradeoffs that a threshold cryptographic scheme  
177 induces across different security properties. A system model is also important to charac-  
178 terize different types of attack that a system may be subject to. Specific attacks in the real  
179 world exploit differences between conventional implementations and their idealized versions.  
180 Threshold schemes can be used to improve resistance against some of these specific attacks  
181 that breach specific security properties (e.g., confidentiality of a key) or sets thereof.

182 An abstract security model is not enough to assess the effects of ~~and on placing~~ a  
183 threshold scheme placed in an adversarial environment. One also needs to characterize  
184 implementation aspects whose variation may affect security. Such characterization helps  
185 distinguish, possibly across different application contexts, the resistance provided against  
186 certain classes of attacks. *To this end, this document proposes that a basis for discussion and*  
187 *comparison of threshold schemes should include the description of several characterizing*

R8: N9

188 *features. These include the types of threshold, the communication interfaces, the target*  
189 *computing platforms, and the setup and maintenance requirements.*

190 The examples in the document illustrate how security properties can vary depending  
191 on high-level features, on assumed attack vectors and on the type of adversarial goals  
192 and capabilities. On one hand, this helps prevent a possible misconception that a higher  
193 threshold directly means higher security. On the other hand, it also intends to convey that  
194 threshold schemes can be used to implement cryptographic primitives in a more secure  
195 way. Altogether, structured security assertions also promote a path for meaningful security  
196 validation of actual implementations.

197 This document considers the benefits of standardizing threshold cryptographic schemes,  
198 possibly along with auxiliary threshold-cryptography primitives. Naturally, there is interest  
199 on threshold schemes for NIST-approved cryptographic primitives. Also of major impor-  
200 tance is the development of corresponding approaches for validation of implementations  
201 of threshold cryptographic schemes. This should be aligned with the current moderniza-  
202 tion process and evolving structure of the testing methodology of the NIST cryptographic  
203 validation programs. Of particular relevance is the development of approaches to enable  
204 automated validation tests with state-of-the-art techniques.

205 The use of well-characterized threshold schemes to implement cryptographic primitives  
206 offers potential security benefits. But what criteria should one use to select from a potential  
207 pool of candidate threshold schemes? What flexibility of features and parameters should a  
208 threshold-cryptographic-scheme standard allow? Should some base primitives be indepen-  
209 dently standardized and/or validated? This document does not offer definitive answers to  
210 these questions. Instead, it motivates the need to develop an objective basis for addressing  
211 them. It also hints at various representative questions to consider, namely about security  
212 assessment, efficiency and applicability, among others.

213 There are important challenges and opportunities related to the standardization of thresh-  
214 old cryptographic schemes. Addressing these may bring about important security improve-  
215 ments to real implementations of cryptographic primitives. Fortunately, there is a plethora  
216 of research work done in the broad area of threshold cryptography, providing useful insights  
217 about possible options, caveats and tradeoffs. Further value can arise from addressing  
218 these challenges with feedback and collaboration from stakeholders, including academic  
219 researchers, industry participants and government representatives.

220	<b>Table of Contents</b>	
221	<b>1 Introduction</b>	<b>1</b>
222	<b>2 Fundamentals</b>	<b>4</b>
223	2.1 Terminology . . . . .	4
224	2.2 Secret sharing . . . . .	6
225	2.3 Secret resharing . . . . .	8
226	2.4 Threshold cryptography . . . . .	9
227	2.5 Side-channel and fault attacks . . . . .	11
228	<b>3 Examples</b>	<b>12</b>
229	3.1 Threshold signature examples . . . . .	12
230	3.2 <del>Examples of side-channel</del> <a href="#">Side-channel</a> attacks and countermeasures . . . . .	15
231	<b>4 Models</b>	<b>16</b>
232	4.1 Security considerations . . . . .	17
233	4.2 Types of attack . . . . .	21
234	4.3 System model . . . . .	24
235	<b>5 Characterizing features</b>	<b>28</b>
236	5.1 Threshold values . . . . .	28
237	5.2 Communication interfaces . . . . .	31
238	5.3 Target computing platforms . . . . .	33
239	5.4 Setup and maintenance . . . . .	37
240	<b>6 Validation of implementations</b>	<b>40</b>
241	6.1 The existing CMVP and FIPS 140-2 . . . . .	40
242	6.2 Integration of threshold cryptographic schemes . . . . .	41
243	<b>7 Criteria for standardization</b>	<b>42</b>
244	7.1 <a href="#">Representative questions</a> . . . . .	42
245	7.2 <a href="#">Standardization at what granularity level?</a> . . . . .	44
246	7.3 <a href="#">Standardization opportunities</a> . . . . .	45
247	<b>8 Conclusions</b>	<b>46</b>
248	<b>References</b>	<b>48</b>
249	<b>List of Figures</b>	
250	1 Illustration of Blakley secret sharing . . . . .	7
251	2 Illustration of share randomization in Blakley secret sharing . . . . .	9
252	<b>List of Tables</b>	
253	1 Representative attack types . . . . .	22

254	2	Characterizing features of threshold schemes . . . . .	29
-----	---	--------------------------------------------------------	----

255 **1 Introduction**

256 Protecting sensitive information from unauthorized disclosure has always been challenging.  
 257 “Two may keep counsel, putting one away,” William Shakespeare wrote in “Romeo and Juliet”  
 258 (1597) [Sha97]. Later, in “Poor Richard’s Almanack — 1735” [Sau34], Benjamin Franklin  
 259 observed that “Three may keep a secret, if two of them are dead.” Today, cryptography is a  
 260 primary means of protecting digital information. In modern cryptography the algorithms  
 261 are well known but the keys are secret. Thus, the effectiveness of encrypting data hinges on  
 262 maintaining the secrecy of cryptographic keys. However, this is difficult in conventional  
 263 implementations, as keys are usually stored in one place on a device, and used there to  
 264 run the algorithm. Devices, much like people, are not completely dependable guardians of  
 265 secrets. Does this mean that keys are the *Achilles’ heel* of cryptography?<sup>1</sup>

266 The localization of a key, for use by an algorithm, is susceptible to enabling leaking  
 267 it out. For example, the internal state of a conventional implementation might be compro-  
 268 mised through a bug such as Heartbleed [DLK<sup>+</sup>14, NVD14], Spectre [KGG<sup>+</sup>18, NVD18a,  
 269 NVD18b]and-, Meltdown [LSG<sup>+</sup>18, NVD18c] and Foreshadow [BMW<sup>+</sup>18], letting an R9: N9  
 270 attacker read private memory locations, including secret keys contained therein. Another  
 271 example is the cold-boot attack [HSH<sup>+</sup>09], which allows recovery of keys from the dynamic  
 272 random access memory (DRAM) of a computer, even seconds to minutes after it has been  
 273 removed from the device. Some attacks inject faults into the computation, for example  
 274 by changing the supply voltage. An example is the “Bellcore” attack [BDL97, ABF<sup>+</sup>03],  
 275 where a fault induces an incorrect computation whose output reveals a secret key. Other  
 276 attacks obtain information through a side channel, such as the execution time, the amount  
 277 of energy it consumes, or the electromagnetic emanations it produces. Many of these fall  
 278 into the category of non-invasive attacks, which can be performed without direct physical  
 279 contact with components within the device. Attacks that exploit leakage of key-dependent  
 280 information can lead to disastrous scenarios in which the master key used to encrypt and  
 281 authenticate device firmware becomes compromised [RSWO17].

282 To counter the inherent security risks of handling secret keys in conventional imple-  
 283 mentations of cryptographic algorithms, technical approaches have emerged that split the  
 284 secret key into two or more shares across different components or parties. For example,  
 285 ~~upon using secret sharing the compromise of one (or more, but not all~~when secret sharing  
 286 is used on the key, the compromise of up to the confidentiality threshold number  $f$  (out  
 287 of  $n$ ) of the shares does not reveal information about the original key. Using appropriate R10: N9  
 288 threshold techniques, the shares can then be separately processed, leading the computation  
 289 to a correct result as if the original secret key had been processed by a classic algorithm.  
 290 The threshold approach can thus significantly increase the confidentiality of secret keys in  
 291 cryptographic implementations.

292 ~~In this report, we focus~~ There is a potential benefit complementary to mitigating single-point-of-failure

<sup>1</sup> Some portions of writing were adapted from text appearing at a previous short magazine article [VMB18].

293 issues in hardware and software implementations. The threshold approach can also enable  
294 decentralization of trust, when delegating the ability to perform some cryptographic operation. R11: C1  
295 This can be useful for higher-level distributed applications, e.g., when the performing of a R12: C1, I2, I5,  
296 cryptographic operation should require agreement by multiple parties. E30, H7

297 This report is focused on threshold schemes applied to cryptographic primitives. In  
298 an  $f$ -out-of- $n$  threshold scheme, some security property is tolerant to ~~the some kind~~  
299 of compromise of up to  $f$  out of  $n$  components in the system. ~~This~~ As a mnemonic, R13: A2, E45,  
300 the symbol  $f$  can be thought of as counting the number of “faulty” (i.e., compromised) E60, F2, K11,  
301 components that can be tolerated. This threshold  $f$  can be specific to some implicit type K12  
302 of compromise, e.g., possibly including cases of crash, leakage, intrusion and accidental or  
303 malicious malfunctioning.

304 In a dual perspective, a threshold can be defined with respect to an operational property.  
305 A  $k$ -out-of- $n$  threshold property denotes that the presence or participation of  $k$  correct  
306 components is required to ensure some correct operation. The relation between the different  
307 thresholds (respectively represented by symbols  $k$  and  $f$ ), e.g.,  $k = f + 1$  or  $k = 2f + 1$ , can  
308 vary depending on the scheme and on the type of compromise and security property.

309 The threshold paradigm brings several security advantages but also some potential weak-  
310 nesses. For example, the use of multiple shares increases the attack surface to encompass  
311 all shares. Thus, the security effect of implementing a threshold scheme depends on an  
312 attack model. It is particularly relevant to consider how difficult may be the compromise of  
313 more than the threshold number  $f$  of components. In some cases, for example with low  $f$ ,  
314 the increased attack surface may enable an attack more efficient and effective than possible  
315 against a conventional (non-threshold) primitive.

316 The threshold concept can apply to security properties of interest beyond the secrecy of  
317 keys. For example, it is useful to enable availability and integrity of computations in spite of  
318 malfunctioning of some of its components. Traditional techniques of fault tolerance often  
319 achieve such resistance when considering random or predictably modeled faults. However,  
320 we are specially interested in resistance against targeted attacks, which can be malicious and  
321 arbitrary. Considering a wide scope of security goals, threshold schemes can exist in several  
322 flavors, depending on the security aspects they address and the techniques used. There are  
323 challenges in ensuring the simultaneous upholding of diverse security properties, such as  
324 secrecy of key material, correctness of outputs and continued availability.

325 In fact, the security impact of a threshold design may be different across different  
326 properties of interest. For example, in some schemes the compromise of one share might not  
327 reveal the original key, but the corruption of a single share (or of a computation dependent on  
328 it) may affect the integrity of the output. These observations highlight the need to look at the  
329 security benefits brought by threshold cryptography as a possible tradeoff across properties.

330 The basic security model for cryptographic algorithms assumes an ideal black box, in  
331 which the cryptographic computations are correct and the internal states are kept secret.

332 For example, such ideal constructs have no side channels that could leak secret keys. This  
333 model contrasts with the reality of conventional implementations, which can be subject to  
334 attacks that exploit differences between the ideal and real worlds. Threshold schemes deal  
335 with some of those differences, by providing tolerance against the compromise of several  
336 components. They may also hinder the exploitation of existing compromises (such as noisy  
337 leakage) from a set of components, e.g., providing resistance against side-channel attacks.

338 A separate analysis of different security properties may lead to some pitfalls. Some  
339 formal models of security are useful to avoid them. The ideal-real simulation paradigm,  
340 common to analysis of secure multi-party computation protocols, combines the notion of  
341 security into a definition of an ideal world. This abstraction captures an intended application  
342 in an ideal world, then allowing security properties to be derived therefrom. Complementary,  
343 a system model is also important to characterize different types of attack that a system may  
344 be subject to. Specific attacks in the real world exploit differences between conventional  
345 implementations and their idealized versions. Some of these may target breaching specific  
346 security properties (e.g., confidentiality of a key) or sets thereof. There is a particular interest  
347 in understanding how threshold schemes can be used to improve resistance against these  
348 specific attacks. It is also relevant to assess potential tradeoffs that a threshold cryptographic  
349 scheme induces across different security properties.

350 There are techniques designed to mitigate foreseen compromises in more complicated  
351 scenarios. For example, verifiable secret-sharing enables detection of misuse of shares by a  
352 shareholder, thereby enabling operational modes that tolerate this kind of corruption. As an  
353 other example, proactive secret sharing can be used to periodically reshare a secret, thereby  
354 periodically reducing to zero the number of compromised shares. However, an abstract secu-  
355 rity model is not enough to assess the effects of ~~and on placing~~ a threshold scheme ~~placed~~-in  
356 an adversarial environment. One also needs to characterize implementation aspects whose  
357 variation may affect security. These include the types of threshold, the communication  
358 interfaces, the target computing platforms, and the setup and maintenance requirements.

R14: N9

359 For example, system models and attack types can differ substantially across different  
360 platforms and communication mediums. It should thus be considered how the components  
361 inter-communicate, and how they can be assumed separate and independent vs. mutually  
362 interfering. In a single device setting, this may involve interaction between different components  
363 within a single chip or a single computer. In a contrasting setting, multiple nodes (e.g.,  
364 servers) may be placed in different locations, communicating within a private network or  
365 across the Internet.

R15: A6, E19

366 Altogether, the security assertions made with respect to an instantiated set of features  
367 provide a path for security validation of actual implementations. Of particular interest are  
368 approaches that enable automated validation tests with state-of-the-art techniques. The  
369 use of well-characterized threshold cryptographic schemes to implement cryptographic  
370 primitives offers potential security benefits. It is thus important to develop objective criteria  
371 for selecting from a potential pool of candidate threshold schemes.

372 **Audience.** This document is targeted, with varying goals, at a diverse audience. Internally  
373 for NIST, the goal is to initiate a discussion about threshold schemes for cryptographic prim-  
374 itives. This motivated the inclusion of representative questions relevant to standardization.

375 The document is also written for people with managerial/policy responsibilities in devel-  
376 opment and/or adoption of cryptographic services and modules. For such an audience, the  
377 document highlights critical aspects of the security of implementations that can be signifi-  
378 cantly affected by nuances in the system model and the employed threshold techniques. Sev-  
379 eral simple examples are provided, including some based on classic secret sharing schemes.

380 The text is also directed to experts in cryptography from academia and industry. For  
381 them, the document is an invitation to engage with NIST in a collaborative effort to resolve  
382 the open questions related to the standardization of threshold schemes for cryptographic  
383 primitives and the corresponding guidelines for implementation validation.

384 It is useful to further clarify one intentional design aspect related to the references to re-  
385 lated work. This document intends to initiate a discussion that may lead NIST to standardize  
386 threshold schemes for cryptographic primitives. For that purpose, we sought to convey in  
387 a balanced way that there are feasible threshold approaches, but without showing particular  
388 preferences. In fact, we specifically opted to avoid an assessment of the most recent works,  
389 preferring instead to exemplify precursory threshold techniques. Therefore, we do not make  
390 an exhaustive analysis and do not try to include the depth and nuances typical of a research  
391 paper or a technical survey. We hope that a thorough assessment of state-of-the-art threshold  
392 approaches can be subsequently performed with an inclusive participation of stakeholders.

## 393 2 Fundamentals

394 ~~[[The subsection “Terminology” was previously the last subsection (2.5) of Section 2]]~~

### 395 2.1 Terminology

396 This document makes use of two dual perspectives of a threshold. In “*f*-out-of-*n*”  
397 the threshold “*f*” denotes the *maximum* number of components that can be *compromised*  
398 (with respect to some implicit security property of the components), while retaining some  
399 (implicit) security property for the global system. Correspondingly, in “*k*-out-of-*n*” the  
400 threshold “*k*” denotes the *minimum* number of components that must remain *uncompromised*  
401 to be possible to ensure some security property of the global system.

R16: N8

R17: A2, E45,  
E60, F2, K11,  
K12

402 We borrow terminology from different research areas, with some overlap, using several  
403 terms that share similar connotations: ~~Sometimes~~, sometimes (but not always) ~~they are~~  
404 ~~interchangeable in the context of *f*-out-of-*n* threshold schemes, where *f* denotes a threshold~~

405 ~~number of components that can be compromised without violating some security property~~  
406 ~~of interest in the overall system~~interchangeable. Some informal correspondences ~~÷~~ follow:

407 • **Active/byzantine/malicious**: characterization of compromised nodes, or of an adversary,  
408 when being able to arbitrarily deviate or induce deviations from a protocol specification.

409 • **Agent/component/node/party/share**: a constituent part of an implemented threshold  
410 scheme, affecting the prosecution of a functional goal (a cryptographic operation, in our  
411 context) to be achieved by a collective of parts; most often used to denote one of the  $n$   
412 parts whose compromise counts towards the threshold  $f$ ; when the context is clear, some  
413 terms can designate parts outside of the threshold composition.

414 • **Aggregator/broker/combiner/dealer/proxy/relay**: an agent with a special role in aiding  
415 the setup, execution and/or maintenance of a threshold protocol; usually not accounted in  
416  $n$ , except if explicitly stated as such (e.g., the case of a primary node).

417 • **Bad/compromised/corrupted/controlled/faulty/intruded**: state of a node, whereby it  
418 departs from an ideally healthy state, and starts being counted towards the threshold  $f$ .

419 • **Client/user**: an agent, not in the threshold set of components, who is a ~~stake-holder~~  
420 stakeholder of the result of a cryptographic computation, typically the requester for that  
421 computation.

422 • **Compromise/corruption/intrusion**: a process by which a node transitions from an  
423 ideally healthy state to a compromised state and/or by which it remains therein.

424 • **Good/healthy/honest/recovered**: ideal state of a node, not yet compromised by an  
425 adversary, but susceptible to attacks.

426 • **Honest-but-curious/Leaky/Passive/Semi-honest**: characterization of compromised com-  
427 ponents, or of an adversary, when the internal state of the former is exfiltrated by the  
428 ~~later~~latter, but without altering the computations and message-exchanges specified by the  
429 protocol.

430 • **Recovery/refresh/rejuvenation/replacement**: transition of a node or nodes from a  
431 (possibly) bad state back to a good state; nuances include update, reversion, change and  
432 reset of internal states, as well as effective replacement of physical components.

433 The above notes simply intend to convey intuition helpful for reading the document. We  
434 do not undertake here the goal of unifying terminology from different areas. Cited references  
435 in the text provide necessary context. The encyclopedia of cryptography and security [TJ11]  
436 and the NIST glossary of security terms [Kis13] provide additional suggestions.

437 **2.2 Secret sharing**

438 Secret sharing is based on splitting the key into multiple shares. For example, to split key  $K$   
 439 into three shares  $K_1$ ,  $K_2$ , and  $K_3$ , we randomly select shares  $K_1$  and  $K_2$  from the same key  
 440 space as  $K$ , and let the third share  $K_3 = K_1 \oplus K_2 \oplus K$  be the one-time pad encryption of  $K$ ,  
 441 where  $\oplus$  is the exclusive OR operation if the keys are bit-strings. No two shares provide any  
 442 information about the secret key — all shares are required to recover  $K$ .

443 The described scheme ~~has a “3-out-of-3” property~~, with  $n = 3$  parties, has a threshold  
 444 property: it is a “ $f$ -out-of-3” scheme with  $f = 2$  with respect to the leakage of any two  
 445 shares alone not giving away information of the original secret key; it is a  $k$ -out-of-3 scheme  
 446 with  $k = 3$  with respect to all three shares being required to recover the key. The  $k$  notation  
 447 is used hereafter for the concrete case of secret-sharing schemes.

R18: A2, E45,  
E60, F2, K11,  
K12

448 More generally,  $k$ -out-of- $n$  secret-sharing schemes can be defined, for any integers  $n$  and  
 449  $k$  satisfying  $n \geq k \geq 1$ . Such secret-sharing schemes were independently developed in 1979  
 450 by Shamir [Sha79] and Blakley [Bla79]. There, any  $k$  parties together can recover a secret  
 451 shared across  $n$  parties, but  $k - 1$  parties together do not know anything about the secret.

452 **Blakley secret sharing.** With the help of Fig. 1, we describe an example of Blakley’s  
 453 scheme for  $k = 2$  and  $n = 3$ , with some simplifications for illustration purposes. The secret is  
 454 the  $x$ -coordinate ( $x_s$ ) of the point  $P(x, y)$  in the two-dimensional plane (see Fig. 1(a)). A non-  
 455 vertical line in the plane is defined as a set of points  $(x, y)$  satisfying  $y = hx + g$  for some con-  
 456 stants  $h$  and  $g$ . If Alice obtains coefficients  $h_A$  and  $g_A$  for some line  $\{(x, y) : y = h_A x + g_A\}$ ,  
 457 containing the point  $P$ , this does not give Alice any advantage in discovering its  $x$ -coordinate  
 458  $x_s$  (see Fig. 1(b)). This is because the definition of the line does not provide any special  
 459 information about any point in the line, i.e. all points in the line (and all  $x$ -coordinates) are  
 460 equally likely. In practice, lines are selected only from a finite space of lines, e.g., with all  
 461 coefficients being integers modulo some prime number  $Q$ , and the lines themselves are finite  
 462 collections of points, e.g., with  $x$  and  $y$  being also integers modulo  $Q$ . The prime modulus  
 463  $Q$  must be larger than the secret  $x_s$  and larger than the number  $n$  of parties.

R19: N8

464 Similarly, if Bob and Charlie obtain coefficients of other lines that pass through the  
 465 same point  $P$ , individually they cannot determine  $P$ . Note that the lines cannot be parallel to  
 466 each other and to Alice’s line. However, any two together — Alice with Bob, or Alice with  
 467 Charlie, or Bob with Charlie — can easily compute  $P$  as the intersection of their lines (see  
 468 Fig. 1(c)). We have thus described a 2-out-of-3 secret-sharing scheme. To build a  $k$ -out-of- $n$   
 469 Blakley scheme for some  $k > 2$ , one considers hyperplanes  $y = h_1 x_1 + \dots + h_{k-1} x_{k-1} + g$   
 470 that intersect in a single point  $P(x_1, \dots, x_{k-1}, y)$  in the  $k$ -dimensional space. ~~The coefficients~~  
 471  ~~$h_i$  are non-zero and  $g$  is an arbitrary constant~~, provided that no hyper-plane is orthogonal  
 472 to the  $x_1$ -axis. Choosing  $n \geq k$  such hyperplanes, one can distribute the corresponding  
 473 coefficients to  $n$  different parties. Then any  $k$  parties together can compute efficiently the  
 474 intersection point  $P$ . ~~The prime modulus  $Q$  must be larger than the secret  $x_s$  and larger than~~

R20: E46

R21: N6

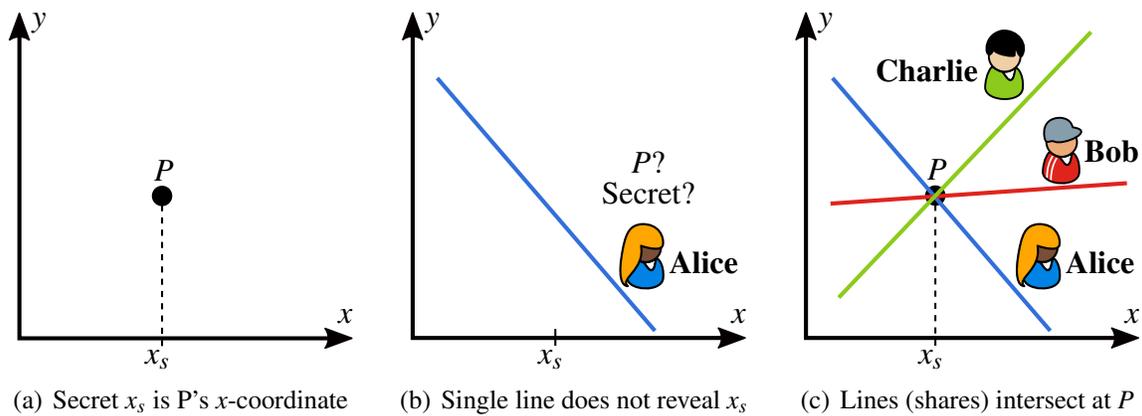


Figure 1. Illustration<sup>2</sup> of Blakley secret sharing

475 ~~the number  $n$  of parties and recover the secret as its  $x_1$ -coordinate.~~

R22: N7

476 Shamir secret sharing. Shamir secret sharing is based on the observation that any set  
 477 of  $k$  distinct points determines completely a polynomial of degree  $k - 1$ . For example,  
 478 consider a set of positive integer coefficients  $c_0, c_1, \dots, c_{k-1}$  and define the polynomial  
 479  $f(x) = c_0 + c_1x + \dots + c_{k-1}x^{k-1}$ . Typically, the secret is the coefficient  $c_0 = f(0)$  and each  
 480 party  $i$  receives as share the point  $(i, f(i))$ , where  $i$  is a positive integer distinct for each party  
 481 (e.g.,  $1, 2, \dots, n$ ). Then, any set of  $k$  parties can reconstruct  $f(x)$ , and therefore compute the  
 482 secret  $f(0)$ , whereas  $k - 1$  parties cannot. All coefficients are based on finite field arithmetic  
 483 defined in terms of a prime number  $Q$ . Since each party must receive a distinct point, and  
 484 that point must not be  $(0, f(0))$ , the modulus  $Q$  must be larger than the number  $n$  of parties.  
 485 The points on the curve are thus defined as  $(x, f(x) \bmod Q)$  and the secret and any other  
 486 coefficient are integers between  $0$  and  $Q - 1$ . This ensures that no information from the  
 487 secret can be recovered from incomplete sets of (i.e., with ~~less fewer~~ than  $k$ ) points on the  
 488 curve.

R23: E48

R24: E49

489 ~~The schemes of Shamir and Blakley's schemes are information-theoretic are information-theoretically~~  
 490 ~~secure, which means that indeed there is no information about the key in a standalone set~~  
 491 ~~of  $k - 1$  shares. This means that the scheme can in practice be used to share very small~~  
 492 ~~secrets (e.g., only a few bits), independently of the application. If, however, the sharing~~  
 493 ~~is applied to a cryptographic key required to be larger than some security parameter, e.g.,~~  
 494 ~~256 bits, then the corresponding prime  $Q$  must be correspondingly large. Alternatively, the~~  
 495 ~~secretsharing could be applied in parallel to independently share portions of a secret. there~~  
 496 ~~is indeed no information about the secret.~~

R25: E50, N9

497 While information-theoretic security may be an advantage, the property requires that each

<sup>1</sup> The humanoid cliparts are from ~~where \* is 2478, 2482 and 2479.~~

<sup>2</sup> The humanoid cliparts are from ~~clker.com/clipart-\*.html, where \* is 2478, 2482 and 2479.~~

498 share is of the same size as the secret, thus meaning that the overall size of all shares is  $n$  times  
499 the size of the secret. In contrast, there are secret-sharing schemes with reduced optimal  
500 size, at the cost of guaranteeing only computational (i.e., cryptographic) security [Kra94].  
501 There, the size of each share can be up to  $k$  times smaller than the size of the secret — this  
502 is ~~especially~~ especially useful if secret sharing is to be used to share large amounts of data.

503 Note: some elements of secret-sharing are standardized by the International Organization R26: N8  
504 for Standardization (ISO) / International Electrotechnical Commission (IEC) [ISO16, ISO17].  
505

### 506 2.3 Secret resharing

507 The need to compute new random shares for the same original secret key often arises in  
508 practice. It may happen that over time some ( $< k$ ) shares are compromised [OY91], thus  
509 creating a need to compute new shares and discard the old ones. Resharing can even be  
510 proactive [HJKY95], e.g., at regular intervals in time and not as a direct response to a  
511 detected compromise.

512 **Resharing in Blakley's scheme.** We continue here the 2-out-of-3 example of Blakley's  
513 scheme, where two parties are required to reconstruct a secret  $x_s$  shared among three parties.  
514 Each resharing of  $x_s$  requires re-randomizing the point  $P$  along the vertical line that defines  
515 the secret. In other words, for each randomization iteration  $r$  a random  $y$ -coordinate  $y_r$  is  
516 sampled, defining a new point  $P_r = (x_s, y_r)$ . Then, the new share (a line) for each party is  
517 also randomized, subject to the constraints that all new lines are non-vertical, intersect at R27: N6  
518 the new point  $P_r$  and are different from one another. With this construction, a single party  
519 (Alice, Bob, or Charlie) still cannot gain any useful insight into the reshared secret  $x_s$ . This  
520 is because at each new resharing  $r$  the point  $P_r$  where the three lines intersect is chosen  
521 randomly in the vertical line that passes through the secret.

522 For visual intuition, we illustrate in Fig. 2 a parametrization based on angles. A line  
523 through a point  $P$  in the plane can be parametrized in terms of its angle  $\omega$ , in the interval  
524  ~~$(-\pi/2, \pi/2]$ ,  $(-\pi/2, \pi/2)$~~ , with respect to the  $x$  axis. Thus, for each resharing iteration R28: N6  
525  $r$  we attribute to each party  $i$  a new random angle  $w_{i,r}$ . ~~An angle is not sufficient to  
526 define a line, so some other reference point is required. The reference cannot be point  $P_r$ ,  
527 since that would reveal the secret, but could for example be the and the  $x$ -coordinate where  
528 the line intersects with the  $x$ -axis. However, this is not even a concern because in practice  
529 the parametrization used is not based on angles, but rather on polynomial coefficients. In  
530 other words, In practice the used parametrization is based on polynomial coefficients, so R29: A4  
531 the share (a line) is ~~not revealed as  $(P, \omega)$  but rather as~~ instead revealed as  $(g, h)$ , where  
532  $y = hx + g$  is the equation that defines the ~~same~~ line.~~

533 For each new iteration  $r + 1$ , one computes a new point  $P_{r+1} = (x_s, y_{r+1})$  and new random

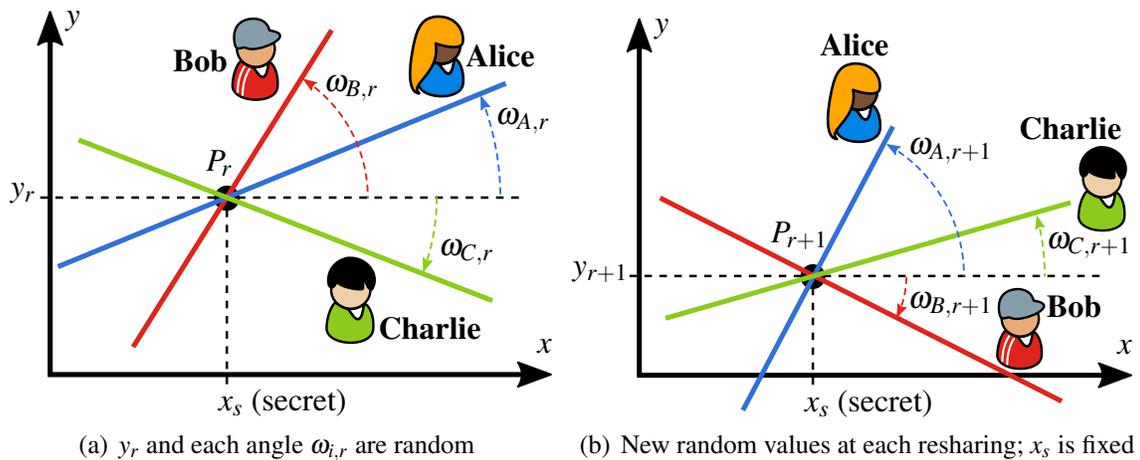


Figure 2. Illustration of share randomization in Blakley secret sharing

534 lines for each party. These lines, passing through point  $P_{r+1}$  correspond to new random  
 535 angles, as illustrated in Fig. 2(b). The dealer (i.e., the party selecting new shares) must  
 536 ensure that the lines of different parties ~~to do~~ not overlap, i.e., that they do not have the same  
 537 angles, and are non-vertical. Concretely, this means that  $\omega_{i,r} \neq \omega_{j,r}$ ,  $\omega_{i,r} \neq \omega_{i,r} \neq \pi/2$  for  
 538  $i, j \in \{A, B, C\}$  and  $i \neq j$ . The generalization to the case  $k > 2$  is as before: the new point  
 539  $P$  would require randomizing  $k - 1$  coordinates, and the resharing would proceed as in the  
 540 initial sharing.

R30: N6

R31: E53

541 **Resharing in Shamir’s scheme.** Share re-randomization can also be done with Shamir  
 542 secret sharing. There, the fixed secret is  $c_0 \bmod Q = f(0) \bmod Q$ . At each random-  
 543 ization iteration  $r$ , one chooses random coefficients  $c_{1,r}, \dots, c_{k-1,r}$  for a new polynomial  
 544  $f_r(x) = c_0 + c_{1,r}x + \dots + c_{k-1,r}x^{k-1}$  satisfying  $f_r(0) = c_0$ . The new shares are then points  
 545 evaluated with  $f_r$ . Concretely, each party  $i$ , for  $i = 1, 2, 3, \dots$  receives  $f_r(i)$  as its new share.

546 **Note:** ~~several elements of secret-sharing are standardized by ISO/IEC.~~

## 547 2.4 Threshold cryptography

548 We take broad input from several research areas with traditionally distinctive names, but  
 549 with a strong relation to threshold schemes. Since we are focused on the implementation  
 550 of cryptographic primitives, we adopt the umbrella term “threshold cryptography” to de-  
 551 note our area of interest. The expression “threshold cryptography” has been traditionally  
 552 used to refer to schemes where some computation is performed over secret shares of in-  
 553 puts [DF90, DSDFY94]. Usually, the setting is such that the shares are used to compute  
 554 something useful, but without being revealed across parties. Often, a main security goal is

555 secrecy of cryptographic keys, but a variety of other security properties, such as integrity  
556 and availability, may also be a motivating drive. Achieving these properties is possible  
557 based on a variety of techniques. For example, integrity may in some settings be enhanced  
558 based on verifiable secret sharing schemes [AMGC85, Fe187] and/or zero-knowledge proofs  
559 [GMR85, BFM88], allowing checking whether shares are used consistently. Specifically, a  
560 threshold scheme can be made robust against adversarially induced inconsistencies in shares  
561 or in related computations, outputting correct results in spite of up to a threshold number  
562 of compromised parties [GRJK00]. While we focus on secure implementations of cryp-  
563 tographic primitives, the actual threshold techniques may also include non-cryptographic  
564 techniques, e.g., simple replication and majority voting.

565 One main area of related research is “secure multi-party computation” (SMPC) [Yao86,  
566 GMW87, BGW88, CCD88]. It allows mutually distrustful parties to compute functions  
567 (and randomized functionalities) of their combined inputs, without revealing the corre-  
568 sponding inputs to one another. This can be useful for threshold schemes even if the  
569 inputs of different parties are not shares of some key and/or if the actual computation  
570 requires interaction between parties. Provided suitable definitions and assumptions,  
571 any cryptographic primitive can be implemented in a threshold manner based on SMPC.  
572 Often this is based on a framework of definitions of ideal functionalities, and protocols  
573 that emulate those functionalities. Nonetheless, some systems may implement threshold  
574 techniques (e.g., secret-sharing, replication) not modeled within an SMPC framework.

R32: E55, I11

R33: E56, J2

R34: E56

575 Threshold schemes also do not encompass all that exists in the realm of SMPC. In  
576 usual SMPC descriptions, the parties themselves are ~~stake-holders~~ stakeholders of the  
577 secrecy of their input and correctness of the output, e.g., in the ~~millionaire’s~~ millionaires’  
578 problem [Yao82] and in secure set intersection [FNP04]. ~~Conversely, threshold schemes are~~  
579 ~~often envisioned at a higher level, where the threshold entity has~~ In threshold schemes for  
580 cryptographic primitives, the nodes within the threshold system can have a neutral interest  
581 for the outcome, and ~~is~~ is in fact just be a service provider (of cryptographic services) to a set  
582 of external users/clients. ~~In other words, threshold schemes do not encompass all that exists~~  
583 ~~in the realm of SMPC, and vice-versa there are threshold schemes not based on SMPC.~~

R35: E56, N9

584 Threshold schemes can also be based on elements from the “distributed systems” re-  
585 search area, where fault and intrusion tolerance are main topics. Common properties of  
586 interest in distributed systems are liveness (making progress even in the face of concurrent  
587 execution/requests) and safety (ensuring consistency of state across multiple parties). Why  
588 would this be relevant for threshold cryptography? As an example, consider implementing  
589 a multi-party threshold version of a full-fledged cryptographic platform. Such a platform  
590 would perform a variety of cryptographic operations, some using secret keys, and based  
591 on requests by users whose credentials and authorization profiles may be updated across  
592 time. Now we could ask: in a setting where *availability* (of cryptographic operations) is  
593 a critical property, and where the system is supposed to operate even in cases of network  
594 *partition* (i.e., even if some parties in the threshold scheme cannot inter-communicate), can  
595 *consistency* (of state, e.g., credentials, across different parties) be simultaneously satisfied

596 under concurrent executions? This is a kind of “distributed systems” problem relevant for  
597 threshold schemes. There are settings [Bre12] where these three properties (consistency,  
598 availability and partition tolerance) cannot be guaranteed to be achieved simultaneously.

## 599 2.5 Side-channel and fault attacks

600 The secrecy of keys can be compromised by the leakage of key-dependent information  
601 during computations. This is possible even without direct physical contact with components  
602 within the device. For example, the time taken, the power consumed, and the electromagnetic  
603 radiation emanated by a device can be measured without penetrating the device enclosure.

604 We will assume that, regardless of whether the computation is in hardware or software,  
605 the device that performs the computation consists of some circuit with wires connecting to  
606 logical gates and memory cells. Then, the attacker’s view of the circuit elements may be  
607 noisy (the *noisy leakage* model [CJRR99]), or the attacker may be limited by the number  
608 of wires of the circuit that it can observe within a certain period of time (the *probing*  
609 model [ISW03]). The noisy leakage model and probing model have been unified [DDF14].  
610 In both models, under some reasonable assumptions on the statistical distributions of side-  
611 channel information, the complexity of a side-channel attack of a suitable implementation  
612 with an  $n$ -out-of- $n$  secret-sharing increases exponentially with the number of shares.

613 As such, side channel attacks on secret-shared implementations become infeasible if the  
614 number of shares is sufficiently high, and is further thwarted when the shares are refreshed  
615 before the attacker can collect enough side-channel information. Further refinements of  
616 the model take transient behavior (“glitches”) of the transistors into account, which can  
617 be handled by Threshold Implementations (TI) [NRR06] or by “lazy engineering” to just  
618 increase the number of shares [BGG<sup>+</sup>14].

619 Besides the aforementioned side-channel attacks, an attacker may also obtain key-  
620 dependent information by injecting a fault into the computation, and then observing the  
621 outputs [BDL97]. To inject the fault, the attacker may, for example, apply a strong external  
622 electromagnetic field. Note that the injection of faults may also introduce errors in the outputs  
623 of the computation, thereby violating the integrity of the outputs. If the threshold scheme  
624 is endowed with the ability to detect which shares have errors, and if the threshold scheme  
625 does not require all shares to be present, it can resist temporary and permanent faults in parts  
626 of the computation. This would provide resistance against a wide range of fault attacks.

627 ~~[[The new subsection “2.1 Terminology” was the old last subsection (2.5) of Section~~  
628 ~~2.]]~~

629 **3 Examples**630 **3.1 Threshold signature examples**631 ~~Basic threshold computation on secret shares.~~632 3.1.1 Basic threshold computation on secret shares

633 ~~Now let us proceed to construct a threshold scheme for digital signatures.~~ First, we recall  
 634 the RSA (Rivest-Shamir-Adleman) signature scheme [RSA78], which defines the public  
 635 key as  $(N, e)$  and the private key as  $d$ , such that  $ed = 1 \pmod{\phi(N)}$ . Here, the modulus  $N$  is a  
 636 product of two large secret primes and  $\phi$  is Euler's totient function. Then, the RSA signature  
 637 for a (possibly hashed) message  $m$  is defined as  $s = m^d \pmod{N}$ . Anyone possessing the  
 638 public key can verify the signature by checking  $s^e = m^{ed} = m \pmod{N}$ . ~~To obtain a-~~

639 Now let us proceed to describe a simple threshold variant of this signature scheme, ~~we~~  
 640 ~~split the-~~ [Fra90, BH98]. We first consider the role of a dealer — someone that, knowing R36: N9  
 641 the secret parameter  $\phi(N)$  and the secret signing key  $d$ , wishes to delegate to other parties  
 642 the ability to jointly produce signatures in a threshold manner. The dealer splits the private  
 643 key  $d$  into three shares  $d_1, d_2$ , and  $d_3$ , such that  $d_1 + d_2 + d_3 = d \pmod{\phi(N)}$ . Now, without  
 644 reconstructing  $d$ , it is possible to first process the message independently using each of the  
 645 shares:  $s_1 = m^{d_1}$ ,  $s_2 = m^{d_2}$ , and  $s_3 = m^{d_3}$ ; and then compute the signature  $s = s_1 s_2 s_3$ . Note  
 646 that this is indeed a valid RSA signature, as  $s_1 s_2 s_3 = m^{d_1 + d_2 + d_3} = m^d \pmod{N}$ . This simple  
 647 threshold RSA signature scheme mitigates the risk of exposing the potentially high-value  
 648 private key  $d$ , which ~~doesn't does not~~ appear in any of the three shares that are used in the R37: N3  
 649 actual computations. Thus, compromising any one of the shares, and even two of them,  
 650 poses no threat of exposing  $d$ . Moreover, frequent updates to the key shares ( $d_1, d_2$ , and  
 651  $d_3$ ) would reduce the window of opportunity for attacks and thereby further reduce the risk.  
 652 Refreshing can even occur after every signature.

653 ~~A threshold scheme.~~654 3.1.2 A  $k$ -out-of- $n$  threshold scheme

655 In the above example, all shares must be present. This might be impractical in situations  
 656 where one or more of the shares become unavailable. For such cases, a  $k$ -out-of- $n$  threshold  
 657 scheme could be used when at least  $k$  shares are available. ~~For RSA signatures,~~

658 But how to generalize from the  $n$ -out-of- $n$  to a  $k$ -out-of- $n$  signature scheme (with  $k < n$ )?  
 659 The needed secret-sharing is no longer a simple additive decomposition into shares (in  
 660 the exponent), and correspondingly the combination into a final signature becomes more

661 complex, namely because the share-holders do not know  $\phi(N)$  (the group order). It is  
662 nonetheless possible to slightly adjust the computation of key shares and signature shares  
663 so that the final combination becomes possible even without knowledge of the secret information  
664 [Sho00]. The secret vs. public knowledge of the order of the underlying group can indeed R38: F4  
665 be relevant in the development of diverse threshold schemes, with some schemes taking  
666 advantage of using groups with publicly known group order (e.g., as in the case of ElGamal  
667 decryption [DF90]).

668 So for RSA signatures one can use a 2-out-of-3, e.g., a 2-out-of-3 secret-sharing scheme,  
669 and a corresponding threshold variant of RSA-. Then, in the case of one share being ir-  
670 recoverably lost or breached, the private signature key  $d$  remains intact, available, and  
671 not breached. This means that one can continue to use the same public key to verify the  
672 ~~signature's correctness.~~ correctness of the signature. R39: N3

673 In contrast, when a conventional implementation is breached, the corresponding pub-  
674 lic/private key pair would have to be revoked and a new pair issued. Typically this also  
675 requires an external certification of the public key by a certificate authority and propagating  
676 it to all relying parties. In addition, a 2-out-of-3 threshold signature scheme becomes more  
677 resilient to future share ~~loses~~ losses if it continuously refreshes the key shares, provided  
678 that at most one is compromised at any given time. Note that in a scheme composed of  
679 three separate conventional RSA implementations with independent keys, refreshing would  
680 require updating the public/private key pairs, along with all entailing inconveniences.

## 681 **Avoiding the dealer.**

### 682 **3.1.3 Avoiding the dealer**

683 In the above descriptions, an implicit trusted party, often called the dealer, knows the secret  
684  $d$  and performs each secret-sharing operation. Particularly, the threshold RSA examples  
685 based on a common modulus  $N$  required the dealer to also know the secret prime factoriza-  
686 tion. ~~Without knowledge of such factorization, it is not currently known how to correctly~~  
687 ~~select such modulus and prove its correctness. Thus, the selection of secrets~~ If needed, the  
688 dealer could, without revealing the secret, prove to each share-holder that  $N$  is indeed a  
689 valid product of two primes, by using zero-knowledge proofs [vdGP88]. Nonetheless, in R40: N8  
690 a dealerless setting a threshold computation of shares of the secret signing key  $d$  would  
691 also require a threshold generation of the public  $N$ , along with a secret sharing of its  
692 factorization. This does not lend itself to a straightforward efficient ~~threshold computation.~~  
693 ~~Nonetheless, such threshold selection, even for RSA keys, can still~~ computation, but can in  
694 general be done based on SMPC protocols [BF97]. ~~By using zero-knowledge proofs the~~  
695 ~~needed property on  $N$  can also be proven without revealing the secret~~ Different RSA-based  
696 threshold schemes can take advantage of specialized solutions, with tradeoffs related to the R41: D3  
697 threshold parameters  $k$  and  $n$  and to properties of the prime factorization.

698 Schemes based on ~~different-particular~~ assumptions can enable a more straightforward  
699 selection and verification of the validity of public elements. For example, this is possible  
700 based on assumptions of intractability of computing discrete logarithms in certain groups  
701 of known order. If the group parameters can be verified as correct in a standalone procedure,  
702 then no one requires ~~knowing-having~~ any secret knowledge about the group. Furthermore,  
703 if the selection is made in a way that avoids the possibility of a trapdoor being known, then  
704 the parameters can be trusted by anyone. The intractability assumption can then, for fixed  
705 security parameters, be accepted for ~~universal-global~~ parameters of a group (e.g., [Ber06]).  
706 In particular, this can facilitate a respective threshold mechanism, so that a secret key never  
707 exists locally at any entity. For example, one can then define a dealer-absent threshold  
708 version of a public key generation (the result of an exponentiation), such that each party  
709 knows one share of the secret key (a discrete logarithm) [Ped91, GJKR99].

R42: N9

R43: E39, F4

710 ~~Other constructions.~~ The same possibilities exist for resharing. In suitable threshold  
711 schemes, the share-holders can perform resharing without a dealer, i.e., interacting to create  
712 new shares for the same secret, without ever reconstructing the secret. The final shares  
713 can even be obtained for new threshold structures (e.g., different threshold and number of  
714 parties) [DJ97].

R44: E54, F3,  
E51

### 715 3.1.4 Other constructions

716 The above examples focused on threshold schemes where the secret-key is shared, and then  
717 a threshold scheme enables a generation of a signature identical to the non-threshold manner.  
718 A feature of those schemes is that the final signature is identical to a non-threshold one,  
719 thereby being inherently efficient in size (i.e., not incurring an increase with the threshold  
720 parameter). Such schemes also have the property that the identities of the signatories remain  
721 secret to the external users interested in verifying the correctness of a signature. However,  
722 some settings may favor the identifiability of signatories, e.g., as an accountability and  
723 credibility feature. Each signatory might also prefer retaining an individual public credential,  
724 not wanting to use a private-key share associated with a common public key. Even in this  
725 setting it is possible to devise short threshold signatures, with size equal to a non-threshold  
726 signature. Concretely, “multi-signature” schemes [IN83, MOR01] enable multiple parties,  
727 with independent secret-public key pairs, to jointly produce a common short signature.<sup>3</sup>

728 A multi-signature scheme can be used as a threshold signature scheme where the  
729 application layer, and possibly the user, has added *flexibility* to define which subsets of  
730 signatories determine a valid signature, i.e., beyond structures defined by a pre-determined  
731 threshold number. For example, a multi-signature may be defined as valid if it contains one  
732 signature from each of three groups of individuals in different roles in an organization. The

<sup>3</sup> These should not be confused with “group signatures” [CvH91], where a member of a group signs a message, while proving group membership but remaining anonymous with respect to its identity within the group.

733 verification procedure then depends on the set of independent public keys. For example,  
734 these schemes can be easily based on Schnorr signatures [Sch90, BN06].

735 To complement the resilience in the face of compromise, signatures can also be im-  
736 plemented with a “forward security” property [And02]. Such schemes can be based on  
737 an evolving private key, while the public key remains fixed, so that even a future key  
738 leakage will not allow the adversary to forge past messages, assuming the signer erases  
739 past keys [BM99]. To some extent, this property has some conceptual similarity to the  
740 refreshing we previously described in the RSA example. This property can be achieved also  
741 for threshold signatures [AMN01], including the case of multi-signatures [SA09].

742 In summary, we showed by examples that “threshold signature schemes” can be based  
743 on secret-shared computation of regular signatures or on multi-signatures, with or without a  
744 dealer, with or without robustness, and possibly with forward security.

745 Several of the exemplified threshold schemes take advantage of group homomorphic  
746 properties. While such properties are not applicable in every cryptographic primitive,  
747 threshold computation can still in general be obtained via secure multi-party computation.

### 748 3.2 ~~Examples of side-channel~~ Side-channel attacks and countermeasures

749 Timing attacks were first presented by Kocher [Koc96], and have been shown to be easy to  
750 perform on a variety of cryptographic algorithms. An advantage of timing attacks is that  
751 no specialized equipment is required. Because they do not require physical access to the  
752 system, they may even be performed remotely over the Internet [BB03].

753 A possible countermeasure against timing attacks is to ensure that the implementation is  
754 “constant time,” that is, that its execution time does not depend on the value of the secret key.  
755 This turns out to be surprisingly difficult for many commonly-used implementations. The  
756 reason is that ~~it may not be sufficient to have~~ having “constant-time” source code, that is,  
757 source code without key-dependent branches or memory accesses [Ber05].-

R45: A5, E57

758 ~~Even worse, may not be sufficient. Indeed,~~ an implementation that is free of timing  
759 attacks on one platform, ~~may be vulnerable on another platform.~~ This can happen, for  
760 example, when source code that contains multiplication operations is compiled with a  
761 different runtime library [KPVV16], or when the same binary is executed on a different  
762 processor [Por18].

R46: N3

763 The execution time of the program, however, is just one example of a side channel.  
764 Implementations in hardware and software may also leak through other side channels, such  
765 as power consumption or electromagnetic radiation. The limitation of the currently-known  
766 countermeasures (such as “constant-time” implementations, dual-rail logic, or electromag-  
767 netic shielding) is that they usually do not get rid of all the leakage, but may still be  
768 vulnerable to higher-order or data-dependent leakages.

769 To protect against side-channel attacks, the framework of threshold cryptography can  
770 provide a promising starting point. If the implementation is split into a number of “parties,”  
771 such that no single party holds the entire secret required to perform the cryptographic  
772 operation, then the leakage of information from only one “party” would not enable a  
773 successful attack on the original secret.

774 However, when all these parties reside on a single chip, we must assume that an attacker  
775 can gain *some* (bounded) information about *every* party. In that case, it may happen that the  
776 threshold cryptosystem only complicates a side-channel attack by a small factor, depending  
777 on the number of parties. For example, the  $n$ -out-of- $n$  threshold block cipher by Brickell et  
778 al. [BCF00] uses the  $n$ -fold composition (or cascade) of a block cipher with  $n$  different keys,  
779 which may slow down power analysis attacks only by roughly a factor of  $n$ .

780 Nevertheless, there exist sound countermeasures against side-channel attacks where the  
781 secret variables are split into shares, such that a threshold number of shares can be used to re-  
782 combine the secret, but fewer shares reveal no information at all. We described the theoretical  
783 foundation of these approaches and their resistance against side-channel attacks in Sec. 2.

## 784 4 Models

785 The basic security model for conventional cryptographic algorithms assumes an ideal black  
786 box, in which the cryptographic computations are correct and all internal states, including  
787 keys, are kept secret. ~~Such ideal constructs would not leak any secret information. This~~  
788 ~~basic model leaves aside the possibility of leakage~~ through side-channels, such as timing R47: N9  
789 and power. ~~In other words, in the ideal black-box the time and energy used for operations~~  
790 ~~would be independent of secrets. This may be due to these parameters not being included~~  
791 ~~in the model, or being assumed independent of the secrets (e.g., being instantaneous or~~  
792 ~~requiring constant time assuming instantaneous or constant-time computations).~~ Under this  
793 assumption, ~~one can reduce~~ the problem of ~~evaluating the algorithm’s security properties~~  
794 ~~to the quantifying a security property of the algorithm can be reduced to the problem of~~ R48: N3  
795 ~~evaluating the~~ complexity of the best-known attack against this model.

796 For example, one can define the security strength, which can also be expressed as bit  
797 strength, of different classes of cryptographic algorithms based on the amount of work  
798 needed to perform a brute-force search of the key in a large space related to the key  
799 size. When the algorithms are implemented in real hardware and software, the black-box  
800 assumption can break down in several ways. For example, bugs in the implementation can  
801 lead to side effects that compromise the secret key, as with Heartbleed. Also, the material  
802 and electromagnetic characteristics of the platforms on which the algorithms run can cause  
803 side-channel information to leak and allow attackers to recover the secret key.

804 The distinction of ideal versus real implementations can yield useful insights into  
805 the assessment of threshold schemes for cryptographic primitives. What are the security

806 advantages and disadvantages of performing separate computations on shares of a key,  
807 compared to conventional implementations that use a single secret key? How can threshold  
808 cryptography mitigate the potentially disastrous consequences that a coding error or a  
809 side-channel leak could have on a conventional implementation?

810 This section considers how a range of applicable scenarios may differently affect a range  
811 of tradeoffs between several security properties. These scenarios depend on adversarial goals  
812 and capabilities, and various properties of the system model. It is important to be aware that  
813 security strengthening and weakening may co-exist. The discussion also precludes the next  
814 section, which motivates the need to describe characterizing features of threshold schemes.

## 815 4.1 Security considerations

816 In a first baseline comparison, a real implementation allows vectors of attack not possible  
817 in an ideal black-box. Once these are identified, one asks how to augment conventional  
818 implementations, in the real world, to improve security. Particularly, *how does a threshold*  
819 *approach affect security, compared to a non-threshold approach?* Perhaps security is  
820 improved if an attacker is limited to not compromising more than  $f$ -out-of- $n$  components  
821 within a certain time interval. Also, as explained in Sec. 3.2, a threshold design may  
822 make it inherently more difficult to exploit existing compromises (such as noisy leakage)  
823 in the set of “parties”. While these intuitions are valuable, we want to enable a more  
824 meaningful formulation and/or validation of security assertions about implementations  
825 based on threshold schemes.

826 Two general metrics of interest are *reliability* and *availability* [Rad97]. We can call them  
827 meta-metrics, since we are ~~specially~~especially interested in considering them to measure  
828 (even when just qualitatively/comparatively) the upholding of concrete security properties  
829 related to implementations under attack. Reliability — probability of not failing a security  
830 goal — is specially suited for cases of “all-or-nothing” security, where the break of a certain  
831 property represents a catastrophic failure. For example, if a secret decryption key is leaked,  
832 then secrecy is lost with respect to the plaintext associated with public ciphertexts, without  
833 anything being able to revert it. Availability — proportion of time during which a security  
834 goal is satisfied — can be used to measure the actual “availability” of a service or property,  
835 e.g., the proportion of cryptographic output produced as intended. These metrics also depend  
836 on the mission time of an application, so it is relevant to consider, for example, resilience  
837 enhanced by *rejuvenating* compromised components back into a healthy state.

## 838 ~~Diverse security properties.~~

### 839 4.1.1 Diverse security properties

840 A threshold augmentation may have different effects across different security properties,  
 841 e.g., confidentiality vs. availability vs. integrity, possibly improving one while degrading  
 842 others. To show the nuances, consider the ~~threshold 3-out-of-3~~ threshold RSA-signature  
 843 scheme described in Sec. 3.1.1, supported on a 3-out-of-3 secret sharing of the key. (Recall  
 844 that, with the notation used here, a 3-out-of-3 secret sharing of the key means  $k = 3$  for  
 845 availability, i.e., three parties (out of  $n = 3$ ) are necessary to produce a signature, and  $f = 2$   
 846 for confidentiality, i.e., any subset of only two parties cannot learn anything about the key.)  
 847 There, each node loses visibility of the original signing key, but retains the ability to in-  
 848 fluence the output of a computation dependent on the key. If a compromised node simply  
 849 refrains from outputting, then it compromises the availability of the signing operation. If  
 850 a corrupted node outputs a syntactically valid but semantically incorrect output share, then  
 851 it may or may not compromise ~~integrity~~ the integrity of the final signature, depending on  
 852 whether or not the mechanism (implicit in the example) responsible for recombining the  
 853 output shares is prescribed or not to verify the correctness of the signature.

R49: A2, E45,  
E60, F2, K11,  
K12

R50: N9

854 In summary, ~~for the example scheme considered~~ even for the considered simple example  
 855 of “3-out-of-3” signature scheme (based on a “3-out-of-3” secret sharing), there are dif-  
 856 ferent compromise thresholds for different properties:  ~~$f_C = 2$  for confidentiality~~; ~~For~~  
 857 example, the compromise thresholds for confidentiality (of the signing key) and availability  
 858 (to produce signatures) are respectively  $f_C = 2$  and  $f_A = 0$  ~~for availability~~; ~~similar to the~~  
 859 underlying secret-sharing scheme. For integrity of produced signatures, the compromise  
 860 threshold is by default also equal to  $f_I = 0$  or  $f_I = \infty$  (depending on the protocol) for integrity.  
 861 , since the described mechanism produces an incorrect signature if one of the output shares  
 862 is incorrect. However, one can also consider an analysis that incorporates the context  
 863 of a signature application where the corruption is detected by a verification against the  
 864 provided plaintext. If the detection of a bad signature prevents an error propagation in the  
 865 application, then the integrity compromise can be disregarded ( $f_I = n$ ) and the problem  
 866 be instead classified as an availability issue ( $f_A = 0$ ). More generally, for a “ $k$ -out-of- $n$ ”  
 867 signature scheme:  $f_C = k - 1$  (for the confidentiality of the signing key);  $f_A$  can depend on  
 868 the scheme, but ideally can be as high as  $n - k$ ;  $f_I$  can depend on the scheme and on the  
 869 application definition of integrity compromise.

R51: A3

R52: A3, E61

870 ~~It is thus conceivable that~~ Based on the above, under certain types of attack ~~, the the~~  
 871 exemplified threshold scheme may, in comparison with the conventional scheme, improve  
 872 the confidentiality of the original key, while degrading the availability and/or integrity of the  
 873 intended output. Particularly, this happens if: ~~compromising the integrity or (when  $f_A = 0$ )~~  
 874 compromising the availability of **one** ( $= 1 + f_A$ ) out of the three nodes in the threshold  
 875 version is easier than compromising the availability of a conventional non-threshold version;  
 876 (when  $f_I = 0$ ) if compromising the integrity of **one** ( $= 1 + f_I$ ) out of the three nodes in the  
 877 threshold version is easier than compromising the integrity of a conventional non-threshold  
 878 version; if compromising the confidentiality in the conventional implementation is easier  
 879 than compromising the confidentiality of **all**  $n$  ( $= 1 + f_C$ ) nodes in the threshold version  
 880 (when  $f_C = n - 1$ ). In some attack/compromise models it may be possible to quantify the

R53: A3

881 likelihood of  $f + 1$  nodes being compromised, e.g., dependent on an attack intensity and  
882 rejuvenation pattern [BB12]. In particular, one may find that under certain models the  
883 threshold property induces less reliability or availability ~~, e.g., if not properly provisioned~~  
884 with rejuvenation ~~techniques.~~ ~~mechanisms.~~ If, for example, nodes are of similar type,  
885 such as several hardware security modules (HSMs) or several virtual machines (VMs) in  
886 different computers and have *diversity* at certain levels (OS, vendor, etc.), and if a constant  
887 rate probability of compromise is plausible for certain attack vectors, then it is possible to  
888 analyze the impact of reactive and proactive rejuvenation.

R54: E14, E16,  
G4

889 Consider the mentioned case with threshold  $f_1 = 0$  for integrity. In a context where  
890 integrity is as important as confidentiality, can the above mentioned scheme still be appro-  
891 priate? Yes, since the difficulty of compromising each property may vary with the conceived  
892 type of attack on the implementation. For example: compromising confidentiality may be  
893 possible by *passively* exploiting side-channel leakage from a set of nodes; compromising  
894 integrity may require actively intruding a node to (maliciously) change an internal state (e.g.,  
895 an incorrect share). Particularly, a security property  $P_1$  having a compromise threshold value  
896  $f_1$  lower than the threshold  $f_2$  of another property  $P_2$  does not imply that  $P_1$  is easier to break  
897 than  $P_2$ . Thus, there may be scenarios justifying a threshold scheme with a high threshold for  
898 some properties, even if with a low threshold (including  $f = 0$ ) for others. Properties with  
899 associated threshold 0 may possibly also be distinctively protected per node, e.g., based on  
900 standard non-threshold techniques, ~~or be dealt with at a different application layer.~~ Also,  
901 as already mentioned with an example for integrity, some properties with a low threshold in  
902 a threshold scheme module may be considered in an adjusted way at the application layer,  
903 if the application can handle the compromise of a property (e.g., integrity) in the threshold  
904 scheme. (Still, the compromise of some properties, such as confidentiality of a key, may  
905 often be undetectable).

R55: N9

906 ~~A word of caution: pitfalls of decoupling security properties.~~

#### 907 4.1.2 A word of caution: pitfalls of decoupling security properties

908 A simplistic decoupling of security properties may lead to pitfalls. An enumeration of  
909 separate security properties (e.g., privacy of input and correctness of output) may sometimes  
910 fail to capture relevant dependencies or other independent properties. A typical example in  
911 cryptography research is related to commitment schemes, useful for auction applications as  
912 follows: first, each agent independently commits to a chosen bid, in a way that *hides* its value  
913 but *binds* the agent to the value; then all agents reveal their bids in a verifiable way, and the  
914 one with the highest bid wins. An over-simplistic analysis of the application could determine  
915 that the commitment would only need to ensure *hiding* and *binding* properties — respectively  
916 mappable to confidentiality and integrity properties. However, this would fail to capture a  
917 needed property of *non-malleability* [DDN03]: upon seeing a commitment from someone

918 else, an agent should not be able to produce a new commitment that commits to a value  
919 related to the originally committed value, and which the agent is able to open upon seeing  
920 the opening of the original commitment. There are hiding-and-binding commitments that are  
921 simultaneously malleable [Ped92], which would be ill-suited to the mentioned application.

922 In contrast to the mentioned pitfall, there are formal methods for defining and proving  
923 security. For example, the ideal-real simulation paradigm [Can01] provides an abstraction  
924 that captures the intended application in an ideal world. Starting with such modeling,  
925 one can then deduce diverse properties, such as confidentiality, integrity and availability,  
926 among others (e.g. non-repudiation, or plausible deniability). If some intended property  
927 is not present, then the specified ideal world is not capturing the intended functionality,  
928 and perhaps a different ideal version should be specified. This formal approach may offer  
929 useful properties, such as composability, allowing upper layer protocols to be analyzed by  
930 replacing the threshold protocol by a corresponding ideal functionality.

931 **Specific attacks.** As The above mentioned considerations are also pertinent when changing  
932 from a conventional scheme to a threshold scheme. The threshold augmentation may R56: N10  
933 require adjusting an ideal functionality and/or adding definitions and security properties.  
934 For example, the communication between components of the threshold scheme may be  
935 subject to attacks/compromises that affect security in a way that is not possible in the  
936 non-threshold version (where the notion of communication between components is not  
937 even applicable). As another example, a security property defined with the help of a “game”  
938 (a game-based definition), where an adversary has some access to an “oracle” (e.g., an  
939 encryption oracle), may have to update the game definition (including the definition of  
940 a success) to account for the possibility of several components being controlled by the  
941 adversary.

#### 942 4.1.3 Specific attacks

943 As just conveyed, there is a phase of security assessment that justifies care about pitfalls  
944 of basing the analysis on a limited number of security properties. In that regard, we assume  
945 as baseline that a conventional <sup>2</sup>-implementation already implicitly satisfies the security  
946 requisites of an intended context. For example, if we discuss a block-cipher or a signature  
947 algorithm, then we assume we are talking of corresponding algorithms already suitable under  
948 some formal model. In other words, the reference conventional system would be secure if its  
949 implementation was not subject to compromise in the real world. It is then that we position  
950 our perspective about threshold schemes in a setting that considers specific attack vectors in  
951 the real world. These attacks, exploiting differences between conventional implementations  
952 and their idealized versions, may sometimes be focused on specific security properties, e.g.,  
953 confidentiality of a key. For possible relations between threshold parameters (e.g.,  $f$  and  $n$ ),  
954 other features (see Sec. 5), and the assumed difficulty to perform exploits (e.g., per node), we

955 consider how threshold approaches can affect (e.g., improve) security properties of interest.  
956 This may include asking how difficult it is to compromise more than  $f$  parties, and/or to  
957 extract meaningful information from leakage collected from a threshold scheme. To be clear,  
958 this is not incompatible with threshold schemes being themselves provably secure within  
959 formal models of security, e.g., within the ideal/real simulation paradigm. Our focus is in  
960 asking how and which threshold schemes may improve security in the real world.

R57: E40, J12–  
J14

961 We have been focusing on attacks against the confidentiality of a key, but attacks can  
962 have other goals. An attack focused on breaking availability can try to accomplish a *denial*  
963 *of service* of some cryptographic operation. The consequences can be catastrophic if the  
964 operation is part of a time-sensitive critical mission. Threshold schemes also provide  
965 tradeoffs for availability. If nodes can only fail by crash, better availability can typically be  
966 obtained by increasing the overall number of nodes  $n$ , while keeping  $k$  constant. However,  
967 if nodes can become malicious, then availability (of correct operations, i.e., with integrity)  
968 requires handling faulty nodes. The cost (in  $n$ ) for handling faulty nodes may vary significantly  
969 based on the ability vs. inability to detect and replace faulty nodes and may impose restrictions  
970 on  $f$  and  $k$ .

#### 971 **4.1.4 Proofs of Security**

972 Proofs of security are essential in state-of-the-art cryptography. Their importance in  
973 supporting proposals of cryptographic schemes is recognized in the “NIST Cryptographic  
974 Standards and Guidelines Development Process” document, published by the Cryptographic  
975 Technology Group in 2016 [Gro16]. These proofs can serve as a guide to follow a logical  
976 path to assess the security of a threshold scheme, being useful to identify assumptions and  
977 attack models on which to base security assertions. This can, for example, be helpful to  
978 compare security of a threshold scheme vs. the security of a corresponding conventional  
979 (non-threshold) scheme. Proofs are characterized by different attributes, e.g.: contextualized  
980 to adversary types (e.g., see Sec. 4.2), security parameters (e.g., computational and/or  
981 statistical) and other characteristics of a system model (e.g., see Sec. 4.3); proving an  
982 enumeration of security properties vs. proving that a scheme (protocol) emulates an ideal  
983 functionality; proving security in a standalone setting vs. establishing security under composition  
984 with other protocol executions; being thorough in modeling the real world vs. omitting  
985 consideration of possible real side-channels. Proofs may also vary with the primitive type  
986 and research area, e.g., single-device setting (e.g., threshold circuits) vs. general SMPC.  
987 Overall, results from state-of-the-art research can provide useful insights on these choices.

R58: G7, I38, J8,  
J9, J10

R59: G6

R60: H6

R61: G7

R62: G5, I6, J7, M3, M4

#### 988 **4.2 Types of attack**

989 Security goals are considered with respect to an adversary, also known as an “attacker”.  
990 When evaluating a proposal for threshold scheme implementation, we would like to have

**Table 1.** Representative attack types

<b>AxisType</b>	<b>Representative question</b>
passive vs. active	Does the attack affect the specified protocol flow?
static vs. adaptive	To which extent are the choices of the attacker based on observations of the protocol execution?
communication interfaces vs. side-channels	Is the attack based on information channels not modeled in the protocol specification?
detectable vs. undetectable	Is the system aware of (e.g., reacts to or logs evidence of) attempted attacks and/or successful intrusions?
invasive vs. non-invasive	Does an attack require physical access to and/or does it affect the physical structure of a device?
threshold-related vs. similar between non-threshold and nodes	Is an attack on the threshold scheme a straightforward generalization (e.g., parallel or sequential attack to nodes) of a possible attack to the conventional implementation?

991 a sense of the range of adversarial scenarios that it may be able to withstand. As a baseline  
 992 to crosscheck security assertions, we consider several attack types, as enumerated in Table 1.  
 993 This is not intended as a full characterization or rigorous taxonomy, but it helps us recall  
 994 and differentiate relevant cases when considering threshold schemes.

995 **Passive vs. active.** A passive attacker (or a passively corrupted node) does not change  
 996 the flow of the prescribed protocol execution, but may gain knowledge of the internal state  
 997 of some participants, as well as read the content transmitted via communication channels.  
 998 In active attacks, some components may be subject to intrusion and behave arbitrarily  
 999 differently from the protocol specification; in the later case, the attacker may also interfere  
 1000 with the communication channels, by altering, dropping and/or reordering messages.

1001 **Static vs. adaptive.** In static attacks, the attack pattern, e.g., the choice of which compo-  
 1002 nents to try to compromise, does not depend on observations of the protocol execution. In  
 1003 adaptive attacks, the attacker can adapt the adversarial actions based on an observation of  
 1004 the protocol flow. For example, a node may be targeted for intrusion upon being elected to  
 1005 a role of *leader* in a phase of the protocol.

1006 **Communication interfaces vs. side-channels.** Some attacks can be perpetrated via regu-  
 1007 lar communication channels, though possibly using specially crafted messages. For example,  
 1008 a corrupted client may send an invalid message to a node of a threshold scheme in order  
 1009 to exploit a buffer-overflow vulnerability. Other attacks can be based on *side-channels*, as

1010 mentioned in Sec. 3.2, taking advantage of an information flow outside the scope of the  
1011 explicitly designated communication interface of the system.

1012 **Detectable vs. undetectable.** Attacks may be detectable (and detected or undetected) or  
1013 undetectable. The latter may happen due to adversaries that are able to bypass possible  
1014 attack-detection mechanisms. They may also result from blatant attacks, if the attacked  
1015 system is nonetheless unprepared for detection. When a system does not detect being under  
1016 attack or having been compromised, it is unable to initiate reactive measures of attack miti-  
1017 gation. It may nonetheless have proactive measures in place, triggered at regular intervals of  
1018 time, e.g., replacing components that might or might not meanwhile have been compromised.  
1019 The prospect of attack detectability may also act as a deterrent against malicious behavior.  
1020 From a different angle: a stealth attack may lead to a detectable compromise/intrusion; a  
1021 detectable attack may lead to an undetected compromise/intrusion.

1022 **Invasive vs. non-invasive.** Another attack characterization relates to the needed proximity  
1023 and interaction between the attacker and the physical boundaries of the attacked system.  
1024 Non-invasive attacks do not require interaction within the physical boundary of the sys-  
1025 tem [ISO12]. Invasive attacks require the attacker to be in the presence of (e.g., “touching”)  
1026 the physical device or be in its immediate proximity. This includes the case of stripping out  
1027 some coating layers of a device, to reach an area of a circuit that can then be directly probed.  
1028 This may also include beaming ultra-violet light into particular zones of a circuit (which  
1029 requires close proximity), to change an internal state (e.g., a lock bit [AK96]) and thereby  
1030 inducing a change of behavior.

1031 **Conventional vs. threshold-related.** While threshold schemes may be designed to miti-  
1032 gate the effectiveness of some attacks on conventional applications, the actual implementa-  
1033 tion of a threshold design may be the cause of new inherent vulnerabilities. For example,  
1034 an attack may be able to exploit some vulnerability in the communication network that  
1035 intermediates several nodes, where such a network would not even exist in a conventional  
1036 implementation. We characterize an attack as threshold-related if the attack vector is in-  
1037 herently allowed by the threshold design. Complementary, there are conventional attacks  
1038 that can be considered similarly with respect to each component of a threshold scheme. In  
1039 the latter case, it is still relevant to consider, for example, if an attacker is able to choose  
1040 whether to attack the nodes/platform in parallel or sequentially.

1041 Tolerance to compromise can be useful even in scenarios of non-intentional adversaries.  
1042 For example, some systems may be constrained to satisfy auditability requirements that  
1043 warrant taking down components for audit. If a service is supported on a multi-party  
1044 threshold scheme with tolerance to compromise, then the audit of components can be done  
1045 without affecting the overall availability.

### 1046 4.3 System model

1047 The goal of this subsection is to convey possible nuances of system models, in order to  
1048 encourage a reflection of different consequences they may induce. Several characterizing  
1049 features of system model for threshold schemes are further discussed in Sec. 5.

### 1050 **Interactions.**

#### 1051 4.3.1 Interactions

1052 For a security assessment, it is relevant to consider the interaction between the threshold  
1053 system and its environment. A threshold system, e.g., a module composed of  $n$  nodes, usu-  
1054 ally interacts with its clients/operators, through a medium of communication. The system  
1055 may also include other interfaces through which a (possibly stealthy) adversary may obtain  
1056 information and/or actively interact with components of the system. Thus, attack vectors  
1057 are not limited just to actual intrusion/compromise of nodes, but also to adversarial effects  
1058 on the environment. For example: corrupted clients may behave maliciously to try to induce  
1059 a denial of service for other clients; an adversary controlling part of the network might be  
1060 able to induce a state of inconsistency across different nodes, even if no node in particular  
1061 can be said to be compromised. We are interested in security properties involving both the  
1062 threshold entity and the complementary environment.

1063 Besides the  $n$  nodes and users/clients, there may also exist special auxiliary components  
1064 with the task of relaying, proxying and/or aggregating messages. Such components, which  
1065 we may call *brokers*, can conceivably be outside of the threshold compromise model (i.e.,  
1066 not accounted in  $n$ ). Particularly, it may be justifiably assumed that a broker does not fail  
1067 within the attack model considered for the other components. For example, a broker may  
1068 be a simple stateless web-redirector, independent of the cryptographic computation needed  
1069 by the threshold components. Conversely, the  $n$  nodes accounted for the threshold may be  
1070 instantiated in a platform more susceptible to certain attacks.

1071 A broker can be used to modularize some concerns, e.g., replacing or substantiating  
1072 usual assumptions, such as the existence of authenticated channels. Depending on the  
1073 communication model, the broker can, for example, broadcast messages from clients to  
1074 all components. At the inter-node level, the broker can be a router at the center of a star  
1075 configuration, substantiating an inter-node (logical) clique model. The broker can also act as  
1076 a mediator between each client and the set of nodes of the threshold scheme, possibly hiding  
1077 from the client the threshold scheme layer. For example, the broker can produce secret shares  
1078 of the client's messages and then only send these shares to the nodes; in the reverse direction,  
1079 it can check consistency, and possibly perform error correction, and aggregate replies from  
1080 a threshold number of nodes, to then just send a consolidated reply to the client. Depending  
1081 on the protocol, the threshold nature can be hidden or not from the client. Even in the broker

1082 case, the threshold nature of the scheme may, as a feature, be intentionally revealed to the  
1083 client. For example, the client may receive a multi-signature enabling non-repudiation of  
1084 the participation of a number of nodes in the production of a response.

1085 The security of a cryptographic service also depends on the communication model. Con-  
1086 ceivably, an attacker may be able to eavesdrop, delay, drop, corrupt and/or forge messages  
1087 in a number of communication channels. A protocol secure in the case of synchronous,  
1088 fail-safe (messages always delivered) and authenticated channels ~~;~~ may become insecure if  
1089 the channel conditions change. Thus, the characterization of the communication model is  
1090 essential to contextualize security claims about a threshold scheme. Main characterizing  
1091 parameters include the existence or lack of synchrony, authentication and encryption. Also,  
1092 the presence of certain trusted components (or trusted setups) may significantly affect the  
1093 capabilities of the system. For example, the existence of trusted clocks may sometimes  
1094 be sufficient to counteract certain difficulties imposed by asynchronous communication  
1095 channels. It is specifically pertinent to justify when the communication medium should be  
1096 protected with some mechanism, such as transport layer security (TLS)~~should be or not be~~  
1097 ~~required for communication, Internet protocol security (IPSec) or others.~~

## 1098 **Identity trust.**

### 1099 **4.3.2 Identity trust**

1100 It is easy to leave implicit certain assumptions about the identities of nodes involved in a  
1101 threshold scheme, but different settings lead to different results. Who decides and enforces  
1102 who the participants (nodes) of a multi-party threshold scheme are? Is the identity of  
1103 each party verifiable by other parties? Is the set of parties constant, does it change in a  
1104 well-defined manner, or is it arbitrarily open to new membership?

1105 In an easy scenario, no new nodes join after the onset of a threshold scheme, and their  
1106 identities remain valid throughout their lifetimes. A *dealer* knowing a secret can define  
1107 the setup configuration, deploying nodes, establishing their identities and possibly even the  
1108 inter-node communication channels. The dealer then distributes shares of the secret and  
1109 delegates the threshold execution of some cryptographic primitive.

1110 A threshold scheme may also be implemented in a setting where the nodes have identities  
1111 tied to public keys within a public-key infrastructure (PKI). The PKI can then support secure  
1112 authentication and communication (e.g., with confidentiality and integrity of content and  
1113 origin) between any pair of nodes. (This assurance assumes that the attacker may control  
1114 the delivery of messages between nodes but cannot prevent nodes from accessing the root  
1115 certification authority.) With PKI-based signatures, a threshold scheme can be designed to  
1116 enable external users to verify that results were indeed obtained upon a threshold interaction.

1117 In a different setting, the initial state of parties might be defined by a joint protocol, e.g.,  
1118 a distributed key generation [Ped92]. The joint computation may yield to every node a share  
1119 of a new secret, possibly along with authentication credentials. This can conceivably be  
1120 used by a certification authority (CA) to generate a new signing key, without ever having  
1121 it available (for leakage) in any localized point. In such case, there is no use for a trusted  
1122 dealer of shared secrets, although the nodes may still have been deployed by ~~a centralized~~  
1123 ~~authority~~the same entity.

R63: N9

1124 Some systems may need or benefit from being dynamic with respect to the number  
1125 of participants in a protocol. This may involve allowing different parties to dynamically  
1126 enter the protocol, thereby making the threshold parameters  $f$  and  $n$  variable (perhaps while  
1127 maintaining a fixed  $f/n$  ratio). What if there is no verifiability criterion for the legitimacy  
1128 of a new intended guest participant? In a Sybil attack [Dou02] a single entity can forge  
1129 multiple entities perceived as valid, thereby easily breaking any fixed threshold ratio  $f/n$   
1130 ( $< 1$ ) of compromisable components. Some mitigation measures may involve enforcing a  
1131 cost of participation per party, e.g., performing some cryptographic puzzle [JB99].

1132 In more controlled settings, there may be a requirement that new parties be able to  
1133 prove belonging to an allowed group. This may be based on a PKI certificate signed by  
1134 an authority. Some scenarios can justify having a dynamic number of parties in an actual  
1135 threshold scheme for cryptographic primitives. This may happen~~for example~~, for example,  
1136 in the case of an implementation with a system of intrusion detection and proactive and  
1137 reactive refreshing of nodes. There may be times when the system must refresh some nodes,  
1138 and due to a high rate of reactive refreshing it may temporarily have no additional nodes to  
1139 join.

1140 ~~Trust between clients and threshold scheme.~~

### 1141 4.3.3 Trust between clients and threshold scheme

1142 We have emphasized the use of threshold schemes as a way to enhance the protection of  
1143 secret keys. But when the threshold system is then used to, say, encrypt or sign messages at  
1144 the request of a client, is there a concern about confidentiality of the plaintext? An intention  
1145 to ensure confidentiality of the plaintext may dictate restrictions on the type of threshold  
1146 scheme and system model. If the plaintext is to remain secret, then the client cannot simply  
1147 send the plaintext in clear to one or several of the nodes. Alternatively, it may for example: (i)  
1148 send it through a trusted proxy that creates and sends a corresponding plaintext share to each  
1149 node; or (ii) it may communicate directly a share to each node; or (iii) it may encrypt shares  
1150 for each node but send them through a single primary node. Each example may be supported  
1151 by a nuanced system model, e.g., respectively (i) the existence of a special trusted component;  
1152 (ii) a communication model where each client can directly communicate with each node;  
1153 (iii) a PKI (or shared symmetric keys) enabling encrypted communication with each node.

1154 We can also consider the assurances that a client would like to receive from a threshold  
1155 scheme operation. We already referred to the possibility of a client receiving independent  
1156 signatures (or multi-signatures) from the nodes. Going further, we can also think of clients  
1157 wanting to obtain assurance of correct behavior by the nodes. This can be achieved, for exam-  
1158 ple, with the support of publicly verifiable secret sharing (PVSS) schemes [Sta96, Sch99].

1159 Another matter related to the relation between users and threshold system is authentication  
1160 and authorization of users. Cryptographic modules often have to support an access control  
1161 mechanism to determine from which users to accept which requests for cryptographic  
1162 operations. Access control can itself be implemented using a threshold approach.

**Distributed  
agreement/consensus**

R64: D8

#### 1163 4.3.4 Distributed agreement/consensus

1164 To explain the importance of defining a system model, we use the distributed agree-  
1165 ment/consensus problem — fundamental in the area of distributed systems — to illustrate  
1166 how varying models can lead to a wide variability of results. This is a relevant problem  
1167 for threshold schemes, namely for certain multi-party implementation settings. The goal  
1168 of *consensus* is to ensure that all good parties within a group of  $n$  parties agree on a value  
1169 proposed by one of the good parties, even if up to  $f$ -out-of- $n$  parties are compromised. For  
1170 example, this may be necessary for letting a multi-party system decide which cryptographic  
1171 operations to perform in which order, when the system receives concurrent requests, possibly  
1172 maliciously delivered, from multiple users.

1173 Results relating  $n$  and  $f$  within this setting include many impossibilities [Lyn89], with  
1174 myriad nuances depending on communication and failure models. In one extreme, the  
1175 problem is unsolvable deterministically in a completely asynchronous setting [FLP85], even  
1176 with (non-transferable) authentication and a single crash-stop process (which can only fail  
1177 by crashing). Yet, realistic subtle nuances of the system model circumvent the impossibility.

1178 For example, the problem is solvable even with Byzantine faults if the processes have  
1179 access to randomness [Ben83, Rab83] or synchronous communication [PSL80, LSP82,  
1180 DDS87]. In those ~~cases~~ settings the number of good components must be larger than  
1181 two-thirds of the total, i.e.,  $k \geq (2n + 1)/3$ , or equivalently  $n \geq 3f + 1$ . ~~If~~ Provided the  
1182 appropriate timing assumptions, if nodes only fail by crash, then a non-crashed simple-  
1183 majority is sufficient, i.e.,  $k \geq f + 1$ , or equivalently  $n \geq 2f + 1$  [Lam06]. In another  
1184 extreme, consensus is solvable even with a single good party if a suitable trusted setup  
1185 can be instantiated to enable transferable message authentication. This is the case when a  
1186 PKI setup enables cryptographic signatures [PSL80], or in some other setups (e.g., reliable  
1187 broadcast and secret channels in a precomputation phase [PW92]).

R65: E64

1188 The discussion above motivates reflecting also on the property of brittleness [Vas15].  
1189 This expresses a degree of susceptibility to a breakdown of the security properties (e.g.,

1190 exfiltration of a key) of a particular algorithm due to errors in the configuration and/or  
1191 input parameters. In other words, one is concerned with the fragility of a system with  
1192 respect to changes in the system model or expected setup. Even if a system has all desired  
1193 properties under a well-defined model, it may be unsuitable for real deployment if it fails  
1194 catastrophically under reasonable variations of the environment. One would typically prefer  
1195 instead some kind of graceful degradation. Also related and pertinent is the consideration  
1196 of how protocols behave differently under different types of attack. Some protocols can  
1197 be characterized by two (or more) ordered thresholds (e.g.  $f_1 < f_2$ ), meaning that desired  
1198 security properties hold while the first threshold is not surpassed, but the security failure  
1199 is not catastrophic while the second threshold is not met [FHHW03]. The thresholds can  
1200 also depend on the type of attackers, and different nodes can be subject to different types  
1201 of compromise.

R66: I7, L2, L3

## 1202 5 Characterizing features

1203 We now provide a high-level structured review of characterizing features of threshold  
1204 schemes, to facilitate the discussion towards criteria for evaluation of concrete proposals.  
1205 We intend to motivate a characterization that helps clarify security tradeoffs when reflecting  
1206 on diverse adversarial models. Put differently, we find that the upfront clarification of certain  
1207 high-level features is important for discussing the standardization and validation of threshold  
1208 cryptographic schemes. Table 2 shows examples of possible representations and attributes  
1209 of characterizing features — see Table 2 — it does not intend to be exhaustive.

R67: E21

### 1210 5.1 Threshold values

#### 1211 ~~A threshold.~~

#### 1212 5.1.1 A threshold

1213 From within a total number  $n$  of components, a “threshold” can be expressed in two ways: a  
1214 minimum required number  $k$  of *good* (i.e., non-compromised) components; or a maximum  
1215 allowed number  $f$  of *bad* (i.e., compromised) components. ~~This dual characterization is~~  
1216 ~~useful and we will use it.~~

1217 ~~The considered type of compromise may vary, but we start by focusing simply on~~  
1218 ~~threshold numbers. In~~ Correspondingly, the dual threshold notation —  $f$  vs.  $k$  — enables  
1219 us to pinpoint each perspective, which can be useful. For example: in some cases, a design  
1220 goal is directly set as the ability to withstand the compromise of up to a threshold number  $f$   
1221 of components. ~~In~~; in other cases, design constraints such as cost may directly limit the

R72: A2, E45,  
E60, F2, K11,  
K12

**Table 2.** Characterizing features of threshold schemes

Feature	Representation	Examples
Threshold type	Threshold numbers of bad ( $f$ ) and good ( $k$ ) nodes	$\max f = 0, \dots, (n-1)/3-1, (n-1)/2-1, n-1$ or $\min k = n, \dots, 2f+1, f+1, 1$
	Variation with security property and attack vector	$(k_{\text{Secrecy}}, k_{\text{Integrity}}) = (1, n), ((n-1)/2, (n-1)/2), \dots, (n, 1)$
	Compromise across nodes	common; independent; sequential
Communication interfaces	Client $\leftrightarrow$ crypto module	broadcast; primary node; secret-sharing
	Inter-node structure	star; clique
	Channel protection	TLS; IPsec; <a href="#">dedicated physical connections; trusted paths [NIS01]; application-level encryption</a>
Target executing platforms	Multiple parties vs. single device	multiple interacting computers; multi-chip in single device; threshold circuit design
	Software vs. hardware	VMs as components; HSM; crypto accelerators; crypto libraries; <a href="#">trusted computing environments</a>
	Auxiliary components	global clock; proxy; combiner; <a href="#">random number generator (RNG)</a>
Setup and maintenance	Bootstrap support	dealer; SMPC
	Rejuvenation modes	reactive vs. proactive; parallel vs. sequential
	Diversity <a href="#">generation</a>	offline pre-computation vs. on-the-fly; unbounded vs. limited set
	<a href="#">Diversity levels</a>	<a href="#">operating system; CA; access control; location; vendor; processor architecture; randomization</a>

R68: E65, I14

R69: E17

R70: N3

R71: Diversity levels: D7 (AC), E16 (OS), E62 (CA), E71 (ref), E72 (vendors).

1222 total number  $n$  of components, which in turn may impose a threshold number  $k$  of good  
1223 components, depending on the protocol and adversarial model.

1224 [As already discussed in Sec. 4.1.1, these thresholds \(of good and bad number of components\)](#)  
1225 [make sense when contextualized \(sometimes implicitly\) to some security property. For](#)  
1226 [example, when referring to a  \$k\$ -out-of- \$n\$  secret sharing scheme the  \$k\$  refers to availability](#)  
1227 [\(minimum number of components necessary to recover a secret\), whereas the compromise](#)  
1228 [threshold  \$f\$  for confidentiality of the secret \(maximum number of components that together](#)  
1229 [cannot recover the secret\) is in that case equal to  \$k - 1\$ . The meanings of good and bad and](#)  
1230 [the corresponding thresholds can vary across different security properties. The threshold](#)  
1231 [symbols  \$k\$  and  \$f\$  can be indexed by the corresponding security property \(e.g.,  \$f\_C\$  vs.  \$f\_I\$  vs.](#)  
1232  [\$f\_A\$ , respectively for confidentiality, integrity and availability\), but we omit indices when the](#)  
1233 [context is clear.](#)

R73: A3, E61

1234 **Relating  $n$  vs.  $f$  and  $k$ .**

1235 **5.1.2 [Relating  \$n\$  vs.  \$f\$  and  \$k\$](#)**

1236 When analyzing proposals for concrete threshold schemes, we intend that the system  
 1237 model be sufficiently characterized to enable determining allowed relations between  $n$  vs.  
 1238  $f$  and  $k$ . ~~We now compare two examples that illustrate how~~ Furthermore, it is important to  
 1239 understand how these thresholds can have an extreme variation across security properties.

1240 ~~In Sec. ?? we already showed how a signature scheme, based on a simple~~ As one  
 1241 example, a  $n$ -out-of- $n$  secret sharing scheme, can have secret sharing scheme has an opti-  
 1242 mal threshold for confidentiality ( $f = n - 1$  ( $f_C = n - 1$ , i.e.,  $k = 1$ )  $k_C = 1$ ) and at the same  
 1243 time a pessimal threshold for integrity ( $f = 0$ ). ( $f_I = 0$ , i.e.,  $k_I = n$ ) and availability ( $f_A = 0$ ,  
 1244 i.e.,  $k_A = n$ ).

R74: A2, E45,  
 E60, F2, K11,  
 K12

1245 For another example, consider a threshold randomness-generator, intended to output  
 1246 uniformly random bit-strings, periodically or upon request. In a particular specification,  
 1247 the output randomness can be defined as the XOR of bit-string contributions from several  
 1248 generators of randomness (the components of the threshold scheme). The output is then  
 1249 uniformly random if at least one (good) contribution is a uniformly random bit-string that is  
 1250 independent of the other contributions. Note that the guarantees for independence are impor-  
 1251 tant but out of scope for this report. Thus, this scheme has an optimal integrity threshold, i.e.,  
 1252  $(k, f) = (1, n - 1)$  ( $k_I, f_I) = (1, n - 1)$ , with respect to guaranteeing the ~~desired~~ uniformly  
 1253 random property of a produced output. However, if an output generation requires the par-  
 1254 ticipation of all components, then the scheme also has the worst threshold for availability,  
 1255 i.e.,  $(k, f) = (n, 0)$  ( $k_A, f_A) = (n, 0)$ , since a single bad party can boycott the output.

1256 ~~The two examples above differ with respect to which properties are optimal vs. pessimal.~~  
 1257 ~~The integrity threshold was pessimal~~ In comparison, the two examples have the same  
 1258 availability thresholds ( $f_A = 0$ ), but different integrity thresholds: pessimal ( $f_I = 0$ ) in the  
 1259 first example and optimal  $f_I = n - 1$  in the second ~~one~~. ~~Alternatively~~ example. Furthermore,  
 1260 confidentiality is a relevant property with optimal threshold ( $f_A = n - 1$ ) in the first example,  
 1261 whereas it is not even ~~considered applicable~~ in the second example. ~~The threshold symbols~~  
 1262  ~~$k$  and  $f$  could be indexed by the corresponding security property (e.g.,  $f_C$  vs.  $f_I$  vs.  $f_A$ ,~~  
 1263 ~~respectively for confidentiality, integrity and availability), but we omit indices when the~~  
 1264 ~~context is clear.~~

R75: N9

### 1265 **Different thresholds for the same scheme.**

#### 1266 **5.1.3 Different thresholds for the same scheme**

1267 We gave examples for how the same threshold scheme may be characterized by different  
 1268 thresholds for different security properties. Going further, the thresholds may vary even for  
 1269 a fixed qualitative property (e.g., confidentiality, or integrity, or availability). Typically, an  
 1270 active/malicious/byzantine adversary induces a lower fault-tolerance threshold (i.e., lower  
 1271 tolerance to compromise), when compared to a passive and/or crash-only adversary. The

1272 same is true for system model assumptions, such as asynchrony vs. synchrony of commu-  
1273 nication channels, and the absence vs. existence of a trusted setup such as a public-key  
1274 infrastructure. The distributed consensus problem in Sec. 4.3.4 shows how a threshold can  
1275 ~~widely vary~~ vary widely depending on the setting.

1276 The determination of relevant threshold values can also depend on the primitives used  
1277 and the application context, e.g., how the actual threshold scheme is used in connection with  
1278 other entities. In some applications, a client can check the validity of signatures obtained  
1279 upon request to a threshold signature module. If a detection of an incorrect signature allows a  
1280 proper reaction, then a threshold signature scheme can be useful even if its integrity does not  
1281 tolerate compromised components (i.e., if  $f = 0$ ). One could then argue that the application  
1282 itself allows a different threshold for integrity. Similar verifiability with respect to decryption,  
1283 or symmetric-key encryption, may be more difficult/costlier, though not impossible. In fact,  
1284 certain threshold schemes can be directly built with a property (often called robustness)  
1285 that prevents integrity violations when up to a threshold number of parties misbehave. For  
1286 example, this can be based on verifiable secret sharing schemes, which allow verification  
1287 of correct use of shares. It can also be based on zero-knowledge proofs of correct behavior.

1288 In the simplest form, a threshold  $f$  is a number that defines a simple partition of subsets,  
1289 distinguishing the set of subsets with more ~~then than~~  $f$  nodes from the remaining subsets. It  
1290 is worth noticing that the concept can extend to more general partitions [ISN89, HM00].

## 1291 ~~Representative questions about a proposed scheme.~~

### 1292 5.1.4 Representative questions about threshold values

- 1293 1. For ~~the desired security properties~~ each desired security property, what are the thresh-  
1294 old values (maximum  $f$  and/or minimum  $k$ ), as a function of the total number  $n$  of R76: N9  
1295 components?
- 1296 2. What envisioned application contexts justify a high threshold for some properties at  
1297 the cost of a low threshold for other properties (or of other mitigation measures)?
- 1298 3. How do threshold values vary with respect to conceivable variations of the system  
1299 model (e.g., synchrony vs. asynchrony, passive vs. active adversaries)?

## 1300 **5.2 Communication interfaces**

1301 The augmentation from a conventional cryptographic implementation to a threshold scheme  
1302 impacts the communication model. Conceivably, a client can now communicate with more  
1303 than one component (hereafter “node”), and the nodes can communicate between themselves.  
1304 In Sec. 4.3.1 we already described several nuances of system model, including synchrony vs.

1305 asynchrony, and the possible existence of a broker. We now briefly describe three nuances  
1306 of communication structures related to clients and nodes.

### 1307 ~~Client to/from primary node.~~

#### 1308 5.2.1 Client to/from primary node

1309 The client may communicate with the threshold scheme via a single contact component.  
1310 When such component is one of the  $n$  nodes of the threshold scheme, we can call it a primary  
1311 node for communication. It relays to all other nodes the communication from the client  
1312 (e.g., a plaintext), and inversely the result (e.g., a signature). For example, it aggregates  
1313 intermediate results produced by other components, to then send a single consolidated reply  
1314 to the client. ~~With a static primary node, the threshold tolerance  $f \geq 1$  would not include~~  
1315 ~~the case of communication.~~ In such a setting the system might, for example with respect  
1316 to availability, not be able to tolerate the failure of the primary node (if this role does  
1317 not change across nodes). But other threshold properties, e.g., confidentiality sustained on  
1318 a secret sharing scheme across all nodes, may remain independent of the use or not of a  
1319 primary.

R77: E70

### 1320 ~~From client to all nodes.~~

#### 1321 5.2.2 From client to all nodes

1322 If the client is aware of the threshold scheme, it may be able to replicate a request across  
1323 all components. A possible advantage is ensuring that all correct components receive the  
1324 same request. Correspondingly, the client may also receive replies from all (or a threshold  
1325 number of) components and only then decide on a final result. In a different implementation  
1326 model, the client can perform secret-sharing on the input and then communicate one share  
1327 per component. This can be used to support confidentiality of the input, e.g., a plaintext  
1328 to encrypt or sign. At the very least, this prevents components from applying corruptions  
1329 dependent on the plaintext value. In the reverse direction, the client can reconstruct (possibly  
1330 with error-correction) an output from a set of replied shares.

### 1331 ~~Inter-node communication.~~

#### 1332 5.2.3 Inter-node communication

1333 In typical threshold schemes, the components have to directly communicate between them-  
1334 selves. (An exception is when the client is the sole ~~intermediator~~ intermediary between  
1335 nodes). The inter-node network structure influences the efficiency and security of commu-  
1336 nication. In a star configuration, a primary node intermediates all communication. In a  
1337 clique configuration (i.e., a complete graph), all nodes are able to directly contact any other  
1338 node. For efficiency reasons, a star configuration may be used for most communication  
1339 and a clique configuration be available for secondary communications. A dynamic selec-  
1340 tion of the primary node (also known as leader) may enable overcoming cases of it being  
1341 compromised [CL02].

#### 1342 ~~Representative questions about a proposed scheme.~~

#### 1343 5.2.4 Representative questions about communication interfaces

- 1344 1. Are clients aware of the threshold nature of the implementation?
- 1345 2. How is the initial request from a client propagated through the set of nodes?
- 1346 3. How can the inter-node communication be compromised?
- 1347 4. How does the client obtain a consolidated reply based on a set of partial results  
1348 produced by a set of nodes?
- 1349 5. How is the logical/physical “boundary” (~~see FIPS 140-2~~ [NIS18c]) of the system  
1350 affected by the existing communication channels?

#### 1351 **5.3 Target computing platforms**

1352 To some extent, the implementation platform can be abstracted from some functional  
1353 properties of a threshold scheme. Yet, there are distinctive platform-related aspects relevant  
1354 for security assessment and validation. We elaborate here on three main instances: single-  
1355 device vs. multi-party; software vs. hardware; and auxiliary components. These aspects can  
1356 affect other features and are relevant for the development of validation profiles.

#### 1357 ~~Software vs. hardware.~~

#### 1358 5.3.1 Software vs. hardware

1359 Cryptography is implemented on a variety of computing platforms. In the early days of the  
1360 modern technological revolution in computing and communications, cryptographic algo-  
1361 rithms were implemented predominantly in hardware. Examples of such embodiments are

1362 the secure phone lines between federal offices in the 1970s. Hardware implementations pro-  
1363 vide a level of isolation of the sensitive cryptographic keys and their utilization in processing  
1364 information, along with storage and management of keys and other sensitive parameters.

1365 It is natural to think of the physical boundary of a dedicated circuit board, a dedicated  
1366 chip, a smart card, or USB key. Thus, one can relate that physical boundary to the ideal  
1367 black box boundary introduced in Sec. 4 and formulate a set of security assertions. This in  
1368 fact is the foundation for FIPS 140-2 [NIS01], which was initially developed for hardware  
1369 cryptographic implementations. This standard contains specific security requirements on  
1370 the physical boundary of hardware modules, namely [in Ref. \[NIS01, Section 4\]](#), which are  
1371 concerned with ensuring the attacker cannot probe the circuitry and extract the keys.

R78: N3

1372 As the adoption of cryptography extended into e-commerce over the Internet, software  
1373 implementations of cryptography emerged and over the years became a widely used embod-  
1374 iment for cryptographic primitives. Software cryptographic implementations on a general  
1375 purpose computer (GPC) are just like any other software component that runs within the  
1376 control of an operating system (OS). GPCs are much more porous (see Sec. 1) and tend  
1377 to provide fewer assurances with respect to the isolation of cryptographic keys and other  
1378 security-sensitive parameters from unauthorized access by other applications running on the  
1379 same GPC/OS platform, or remotely through the network interfaces of the platform. Corre-  
1380 spondingly, these software modules are subject only to a subset of the security requirements  
1381 [in ~~described in Ref. \[NIS01\]~~](#) and are limited to a lower level of security assurances they can  
1382 claim to deliver.

R79: N3

1383 Given this historical context, the distinction of hardware vs. software in FIPS 140-2  
1384 comes from the difference in isolation that the approaches provide, and is not directly related  
1385 to the manner in which the computation is performed. Note, for example, that ~~a Hardware~~  
1386 ~~Security Module (HSM) an HSM~~ might actually contain an embedded microcontroller that  
1387 performs the cryptographic computation in *software*. Also, some hardware platforms such  
1388 as a Field-Programmable Gate Arrays (FPGAs) can be “reprogrammed,” a property that  
1389 was historically reserved for software implementations. For the sake of readability, we will  
1390 assume a more “traditional” separation between hardware and software, focusing primarily  
1391 on the isolation properties, rather than on different types of computing platforms.

R80: N3

1392 The hybrid approach to cryptographic implementations aims to benefit from the flex-  
1393 ibility in software and the isolation and/or acceleration in hardware. Here a portion of  
1394 the implementation is in software executing on a GPC/OS platform and another portion is  
1395 executing on a dedicated HSM attached to the same GPC. Examples of such modules are the  
1396 Trusted Platform Module (TPM) [Mor11], or the cryptographic extensions of standard ~~CPU~~  
1397 ~~instruction sets~~ [Central Processing Unit \(CPU\) instruction sets](#), such as the ~~SGX~~ [Software](#)  
1398 [Guard Extensions \(SGX\)](#) instruction on Intel platforms [Int18], [and](#) the TrustZone technol-  
1399 ogy on ~~ARM~~ [Advanced RISC Machine \(ARM\)](#) processors [ARM18]. These modules can  
1400 also be used as secure sub-components within a hybrid fault model. The “secure” compo-  
1401 nents have a more restricted mode of compromise (e.g., only by crash), thereby enabling

R81: N3

1402 better thresholds for byzantine fault tolerance of a distributed system composed also of  
1403 larger and less secure components [VCB<sup>+</sup>13, BDK17].

1404 In some cases, a specific cryptographic primitive is implemented partially in software  
1405 and partially in hardware. For example, an ~~asymmetric~~ RSARSA signature algorithm may  
1406 be implemented in such a way that the modulo exponentiation is executed in hardware but  
1407 the required padding of the data is implemented in software. In other cases, an entire suite  
1408 of fully implemented cryptographic primitives is implemented in an HSM and used by a  
1409 software component through application programming interfaces (API).

1410 The hybrid approach offers important security advantages for implementing crypto-  
1411 graphic primitives and key management in isolation, as well as performance improve-  
1412 ments. For example, a hybrid implementation could potentially mitigate cold-boot at-  
1413 tacks [HSH<sup>+</sup>09], which allows keys to be recovered in seconds or even minutes after it has  
1414 been removed from the device. Cold-boot attacks typically assume that the keys are stored  
1415 in the virtual memory of the operating system, and might therefore be moved into DRAM.  
1416 An HSM could mitigate this attack by ensuring that keys never leave the HSM.

1417 Another reason to delegate the execution of cryptographic primitives to dedicated  
1418 hardware is for performance improvement. An example of this is the AES extension on  
1419 Intel [Gue09] and ~~AMD~~ Advanced Micro Devices (AMD) CPUs [AMD12]. HSMs offer  
1420 similar acceleration benefits.

R82: N3

1421 **Single-device vs. multi-party.**

### 1422 5.3.2 Single device vs. multi-party

1423 When a threshold scheme is developed to enable tolerance to the compromise of several  
1424 components, it is intuitive to think of a set of interacting parties (also known as nodes or  
1425 devices). For example, a *multi-party* threshold setting can be composed of  $n$  computers  
1426 communicating over the Internet, or  $n$  hardware security modules (HSMs) connected via a  
1427 private network, or  $n$  virtual machines (VMs) running within the same hardware machine.  
1428 The connectivity may be dynamic, with the components being possibly replaceable for  
1429 testing, updating and patching. In a multi-party computation, the nodes may be separated  
1430 by a network, possibly asynchronous, inherently outside of the control of the threshold  
1431 scheme. For testing and validation, the tester/validator might not be able to simulate a  
1432 realistic communication medium between multiple parties.

1433 In contrast to the alluded multi-party systems, we also consider “single device” settings.  
1434 Main distinctive aspects include, typically, a somewhat rigid configuration of components  
1435 and a well-defined physical boundary. If the device is a hardware circuit, then in most  
1436 cases the connections between inner wires and gates are fixed throughout the life of the  
1437 device. However, there are technologies that actually allow even those components to be

1438 adapted across the lifetime of the device, e.g. FPGA. Communication synchrony between  
1439 components is often expected and achieved. Threshold schemes are applicable to the single-  
1440 device setting by means of an inner threshold design. There, the inputs and outputs of a  
1441 threshold circuit become encodings (e.g, sets of secret shares) of the inputs and outputs of  
1442 the conventional (non-threshold) circuit. For confidentiality, the threshold property may be  
1443 that no isolated subset of up to  $f$  wires in the threshold circuit contains information about  
1444 any bit that would be flowing in the original circuit. A main application of this design is  
1445 providing increased resistance against certain side-channel attacks [NRR06].

1446 There is flexibility in distinguishing, and identifying similarities, between multi-party  
1447 and single-device scenarios. For example, we could imagine the physical components within  
1448 a device with a threshold design to be multiple “parties”. Conversely, a single-device may  
1449 indeed not have any redundancy of hardware components, and yet a threshold scheme be  
1450 applied by means of repeated executions of an algorithm. The value of distinguishing the  
1451 platforms is in facilitating a categorization of aspects that may justify different standard-  
1452 ization and/or validation profiles. For example, in a multi-party setting it may be easier to  
1453 isolate, replace and test corruption of a singular component, for the purpose of validating  
1454 properties of an implementation. In some single-device cases, it may be infeasible to achieve  
1455 complete separation of components to test their individual correctness.

1456 ~~Auxiliary components.~~

### 1457 5.3.3 Auxiliary components

1458 Threshold schemes may require essential components beyond those accounted in  $n$ . To  
1459 use a distinctive term, we call them *auxiliary* components. These may include ~~for example~~  
1460 , for example, a trusted global clock, a proxy, a common random (or pseudo-random)  
1461 bit generator, a combiner of information from components. Having a threshold-scheme  
1462 characterization that acknowledges these components enables a better system model for  
1463 security assessment. For example: a trusted (assumed trustworthy) clock may be what  
1464 enables synchrony in a system model, which in turn can influence the threshold and the  
1465 protocol; the interaction with a trusted random number generator may be necessary to take  
1466 advantage of the threshold design of a circuit based on secret-sharing; we have also already  
1467 given examples of how the auxiliary components may affect the inter-node and the client-  
1468 node communication interfaces. The auxiliary components may have their own compromise  
1469 model, and their testing and validation is also needed when testing and validating a threshold  
1470 system. Yet, it is foreseeable that a great deal of analysis about the auxiliary components  
1471 can be modularized away from threshold-related arguments.

1472 ~~Representative questions~~

### 1473 **5.3.4 Representative questions about computing platforms**

- 1474 1. If a proposed threshold scheme is devised for a “single-device” setting, what can go  
1475 wrong if its components are instead separated and communicate over the Internet?
- 1476 2. Which parts of the logical boundary of the threshold system do not correspond to a  
1477 physical boundary, as verified by the system developer or deployer?
- 1478 3. Is the system simply developed at the software layer, or are there software components  
1479 tied to particular hardware components?
- 1480 4. Which auxiliary components support the threshold scheme but have a failure model  
1481 different from the one applied to the threshold nodes?

### 1482 **5.4 Setup and maintenance**

1483 In some settings a threshold scheme can be implemented from scratch as an alternative to  
1484 a construction with a single point of failure. In other cases the starting point is exactly an  
1485 existing single-point-of-failure entity, which is intended to be redesigned as a threshold  
1486 system. To compare the effects from the change, we should consider how the system is  
1487 bootstrapped, including “who” deploys the nodes, and their initial states. Also relevant is the  
1488 setup of the communication network and continued maintenance of the system, including  
1489 during detection and/or recovery of compromised components.

### 1490 ~~Dealer vs. dealer-free setup.~~

#### 1491 **5.4.1 Dealer vs. dealer-free setup**

1492 In secret sharing, a “dealer” is an entity, possibly outside the failure model of the threshold  
1493 scheme, that knows a secret and “deals” shares of it to the nodes of the threshold scheme. In  
1494 a possible scenario, a key holder in a safe environment deals shares of a long-term signature  
1495 key to nodes that operate in a threshold manner in a less-secure environment. The role  
1496 of a dealer is not necessarily limited to applications related to secret keys. As a practical  
1497 example, a setup phase can also consist of a trusted party generating and secret sharing  
1498 so-called “Beaver-triplets” — triplets of field elements (possibly bits) where the third is  
1499 the product of the first two. The pre-processing of these triplets enables a very-efficient  
1500 execution of certain secure computation protocols [Bea92].

1501 In a setting with a dealer, it is relevant to consider the extent to which the protocol  
1502 security withstands or breaks in the presence of misbehavior by the dealer [BR07]. Some  
1503 protocols can be made secure against an untrusted dealer, with respect to integrity, if  
1504 the protocol enables parties to verify correctness of the distributed parameters. Other

R83: K2-K6

1505 protocols may have security hinge on an assumption of a trusted dealer. Depending on the  
1506 functionality, there may exist tradeoffs between efficiency and the property of supporting  
1507 the malicious dealer.

R84: K6

1508 **Rejuvenation of nodes.**

#### 1509 **5.4.2 Rejuvenation of nodes**

1510 It is desirable that compromising  $f$ -out-of- $n$  nodes in a good threshold scheme is not easier  
1511 than compromising 1-out-of-1 in a conventional scheme. But is such property inherently  
1512 guaranteed if  $f > 0$  and if the process of compromising each node is independent? Not  
1513 necessarily, even if the compromise of a node requires an independent exploitation effort  
1514 (e.g., time, computation) per node.

1515 If nodes of a threshold system can only transition from an uncompromised to a com-  
1516 promised state, then the system may be less secure under certain attack vectors. This may  
1517 be due to an increased attack surface, a sufficiently low  $f/n$  ratio and a sufficiently high  
1518 mission time. This is a well-known result in fault tolerance, as may happen in a basic  
1519 triple-modular-redundancy design [KK07]. One may also consider adversarial scenarios  
1520 that induce a probability rate of a node under attack becoming compromised [OY91]. To  
1521 counteract these transitions, it is possible, and in many ~~cases~~-settings essential, to imple-  
1522 ment recovery/replacement/rejuvenation of nodes that can bring nodes back to a “healthy”  
1523 (uncompromised) state. There is a plethora of possible rejuvenation modes, e.g., reactive vs.  
1524 proactive, parallel vs. sequential, instantaneous vs. delayed, stateless vs. stateful, etc.

R85: N3

1525 If a compromise is detected, then the corresponding node should be reactively replaced  
1526 by a healthy version, lest the system eventually converges to all nodes being compromised.  
1527 If the compromises are not detectable but are nonetheless conceivable, then a proactive  
1528 recovery should take place. In the threshold signature scheme from Sec. 3, the resharing of  
1529 the secret key constitutes a parallel rejuvenation of nodes. If there is no persistent intrusion,  
1530 and the number of compromises never exceeds the allowed threshold, then the resharing  
1531 brings the whole system back to a pristine state, with all nodes healthy.

1532 The rejuvenation feature brings along a whole new set of considerations, possibly affect-  
1533 ing security in non-trivial ways. If the nodes need to be stateful (i.e., hold state about the  
1534 application), then newly inserted nodes need to be consistently updated, which requires spec-  
1535 ification as a sub-protocol. The rejuvenation of a previously compromised node may need to  
1536 diversify some component, to prevent re-exploitation of the same vulnerability [KF95]. The  
1537 diversification operation may have its own requirements, possibly requiring pre-computation  
1538 vs. being generated on-the-fly by some sampling procedure.

1539 In some protocols a rejuvenation may have to take place in parallel, e.g., such as the  
1540 already discussed example of updating key shares, with all online parties being rejuvenated

1541 simultaneously. In other cases, rejuvenations may occur sequentially, replacing/recovering  
1542 each node at a time, ~~especially~~ especially if the process involves a long ~~down-time~~ downtime.  
1543 Many of the considerations pertinent to the initial setup of a threshold system are also  
1544 applicable to the rejuvenation context. For example, is there a “dealer” responsible for  
1545 setting up the full state of a rejuvenated node or should the state be updated by the set of  
1546 online nodes?

R86: N3

1547 If a threshold scheme is based on electing a primary node, what happens when the  
1548 primary node is the one in need of replacement? If a scheme allows reactive and proactive  
1549 rejuvenations, can an attacker take advantage of knowing the schedule/ordering of the  
1550 proactive rejuvenations? What happens if the regular threshold scheme performs correctly in  
1551 an asynchronous environment, but the recovery procedure requires synchrony? Not handling  
1552 asynchrony in recovery procedures may hide subtle problems [SNV07]. If the regular  
1553 threshold scheme requires only a simple honest majority, but the corresponding rejuvenation  
1554 mechanism requires a 2/3 honest majority, then the threshold properties are also affected.

1555 **Levels of diversity.** A-

### 1556 5.4.3 Levels of diversity

1557 Intuitively, a main motivation for threshold schemes ~~, as an intuitive way is~~ to improve  
1558 security by withstanding the compromise of some nodes.<sup>4</sup> Yet, a standalone characterization  
1559 of threshold values does not say anything about the difficulty of compromising the threshold  
1560 number  $f$  of nodes. Consider the case of a common vulnerability, i.e., common across all  
1561 nodes (e.g., a bug in a common operating system). Once the vulnerability is discovered, an  
1562 adversary might be able to exploit it with negligible cost to compromise all nodes. In this  
1563 example, this would then be “as easy” as compromising a conventional scheme with the  
1564 same vulnerability.

R87: N3

1565 Consider an example where all nodes are symmetric with respect to the threshold pro-  
1566 tocol, i.e., all implement the same functionality. One can then imagine all nodes being  
1567 implemented in the same manner, say, the same software, possibly containing a common  
1568 vulnerability. Conversely, each node can also be programmed for the same functionality  
1569 via different software versions [CA78]. In practice, common vulnerabilities may occur at  
1570 multiple levels where the set of nodes is homogeneous, e.g., operating system, network pro-  
1571 tocol, hardware design, physical location, password. Diversity may be implemented across  
1572 space (i.e., across the components within a threshold protocol) and time (i.e., replacements  
1573 and executions across time). In the multi-party case, rejuvenation can happen by actually  
1574 replacing a physical node by a new one. In certain single-device settings, rejuvenation might  
1575 be limited to refreshing randomness, while the actual hardware structure remains fixed. In a

<sup>4</sup> We also bear in mind the possible mapping of threshold properties into side-channel resistance properties.

R88: Diversity levels: D7 (AC), E16 (OS), E62 (CA), E71 (ref), E72 (vendors).

1576 software setting, rejuvenation may correspond to replacing a virtual machine, or changing  
1577 some randomness used when compiling a software version. At some levels, there may be  
1578 a small set of variants ~~;~~(e.g., operating systems), whereas others (e.g., passwords) are  
1579 impossible to replace.

1580 The use of diversity is a longstanding practice of traditional fault-tolerance, but its use for  
1581 security is more intricate [LS04]. Implementation-wise, multiple levels of *diversity* (among  
1582 other properties) may be required to reduce the possibility of common vulnerabilities [SZ05]  
1583 and to substantiate an assumption that compromising more nodes is more difficult than  
1584 compromising fewer nodes. A fundamental difficulty is that the level of effort used by an  
1585 attack vector may be unpredictable until the attack takes place.

#### 1586 **Representative questions.**

#### 1587 **5.4.4 Representative questions about setup and maintenance**

- 1588 1. Can a threshold scheme be bootstrapped in both dealer and dealer-free manners?
- 1589 2. What levels of diversity are envisioned to deter common-mode failures?
- 1590 3. What dependency of compromise exists across nodes, for envisioned attack vectors?
- 1591 4. ~~Does the sub-protocol for handling rejuvenations interfere with the system availability?~~  
1592 Does the sub-protocol for handling rejuvenations interfere with the system availability?

## 1593 **6 Validation of implementations**

### 1594 **6.1 The existing CMVP and FIPS 140-2**

1595 Governments recognize cryptography's important role in protecting sensitive information  
1596 from unauthorized disclosure or modification, and tend to select algorithms with well-  
1597 established theoretical security properties. For example, US and Canadian federal agen-  
1598 cies must use NIST-defined cryptographic algorithm standards to protect sensitive data  
1599 in computer and telecommunications systems [tC96]. They must also use only validated  
1600 cryptographic implementations, typically referred to as modules.

1601 As we have pointed out, the correct and bug-free implementation of a cryptographic  
1602 algorithm and the environment in which it executes are also very important for security. To  
1603 assess security aspects related to real hardware and software implementations, NIST estab-  
1604 lished the Cryptographic Module Validation Program (CMVP) [NIS18c] in 1995 to validate  
1605 cryptographic modules against the security requirements in Federal Information Processing  
1606 Standard (FIPS) Publication 140-2 [NIS01]. The CMVP leverages independent third-party

1607 testing laboratories to test commercial-off-the-shelf cryptographic modules supplied by  
1608 industry vendors.

1609 FIPS 140-2 is a standard defined as a system of conformance security assertions. The  
1610 security assertions in the standard cover a wide range of cryptographic primitives imple-  
1611 mented into various types of physical embodiments called cryptographic modules. The  
1612 security assertions are grouped into sets, one for each security level. FIPS 140-2 defines four  
1613 security levels for cryptographic modules. Depending on the type of technology used for  
1614 a particular module, e.g. software or hardware, the standard defines a subset of applicable  
1615 security assertions that the module must meet for a chosen security level and module-specific  
1616 functional capabilities. In turn, the cryptographic primitives approved by NIST and adopted  
1617 in FIPS 140-2 through Annex A for use in cryptographic modules are also specified as  
1618 sets of conformance security assertions. This allows the CMVP to work with a reasonably  
1619 constrained and well-defined set of security assertions that can be validated.

1620 The Common Criteria [Com17] follows a contrasting approach, where one is allowed  
1621 to define a unique set of security assertions for a target component, often referred to as a  
1622 target of evaluation (TOE). The goal of the Common Criteria certification then is to evaluate  
1623 the correctness of the specific security assertions claimed by the TOE. The evaluation is  
1624 typically much less structured than the validation process in FIPS 140-2, takes longer time  
1625 and requires substantially higher expertise from the evaluators and validators.

## 1626 6.2 Integration of threshold cryptographic schemes

1627 When we consider standardizing threshold cryptographic schemes for approved NIST cryp-  
1628 tographic primitives, we intend to pursue the approach of conformance security assertions,  
1629 similar to the approach taken for the cryptographic primitives and modules.

1630 FIPS 140-2 already has security requirements for secret sharing applied to cryptographic  
1631 keys. Section 4.7.4 of the standard defines security requirements for split-knowledge  
1632 procedures for security levels 3 and 4, stipulating that “*if knowledge of  $n$  key components*  
1633 *is required to reconstruct the original key, then knowledge of  $n - 1$  components provides no*  
1634 *information about the original key, other than the length.*” This can for example be satisfied  
1635 by implementations of the Shamir and Blakley secret sharing schemes mentioned in Sec. 2.2.

1636 The above-mentioned provision in FIPS 140-2 refers only to secret-sharing and by itself  
1637 does not ensure that keys are never recombined when needed by an algorithm, which is  
1638 a main subject of threshold schemes for cryptographic primitives. That provision is thus  
1639 insufficient to accommodate the plethora of threshold considerations that have been pointed  
1640 out in this report. Generally speaking, the process towards standardization of threshold  
1641 schemes may involve reconsidering the adequacy of the validation requirements and where  
1642 necessary devise new or complementary requirements.

R89: G3,I7,M6

1643 As technology progresses and cryptography becomes ubiquitous in the federal informa-

1644 tion infrastructure, the number and complexity of modules to be validated increases. This  
1645 makes it increasingly difficult to detect at validation stage all possible defects that might  
1646 compromise security. This is one more reason to consider the potential of threshold cryptog-  
1647 raphy in avoiding single points of failure in real implementations. However, similarly to  
1648 conventional cryptography, the security of the threshold cryptographic implementation may  
1649 also be impacted by defects introduced as a result of human errors or unsafe optimization  
1650 by the tools used to compile or synthesize the implementation. Thus, it is important to  
1651 ensure that the algorithms supporting threshold cryptography are theoretically secure, and  
1652 to verify that they ~~’ve~~ have been implemented correctly. The definition of guidelines would  
1653 help develop a structured process of formulating and validating security assertions about  
1654 threshold cryptographic implementations.

R90: N3

1655 One additional challenge is to enable ways to validate those assertions in an automated  
1656 fashion. NIST is working with the industry to rebuild its cryptographic validation programs  
1657 and improve the efficiency and effectiveness of cryptographic module testing in order to  
1658 reduce the time and cost required for testing while providing a high level of assurance for  
1659 Federal government consumers. As the NIST cryptographic validation programs evolve,  
1660 the adoption of new cryptographic technology into them should target the future structure  
1661 and mechanisms for testing and reporting results [NIS18b]. The current project includes an  
1662 Industry/NIST collaboration website for automated validation of cryptographic algorithms  
1663 (ACVP) and cryptographic modules [NIS18a, NIS18b].

1664 It is encouraging to note that automated methods for validating protocol implementations  
1665 have emerged recently (e.g., [CHH<sup>+</sup>17, BBK17, DLFK<sup>+</sup>17]). This experience may be  
1666 useful to leverage for the protocols involved in threshold cryptographic schemes.

R91: E11

## 1667 **7 Criteria for standardization**

1668 Active research over the last few decades has resulted in a substantial body of literature on  
1669 threshold cryptographic schemes. Usually there are tradeoffs of threshold values for different  
1670 security properties, potentially depending on the application context and system model.  
1671 With appropriate caution, threshold cryptography offers a great potential for strengthening  
1672 the security of cryptographic implementations. But what criteria should one use to ask for  
1673 and select from a potential pool of candidate threshold cryptographic schemes?

### 1674 **Some representative questions.**

#### 1675 **7.1 Representative questions**

1676 We intend this document to promote the development of criteria for evaluation of proposals  
1677 of threshold cryptographic schemes. Here we list representative questions likely to induce a  
1678 discussion about this:

1679 1. Characterizing features

R92: N4

- 1680 (a) Are the *characterizing features* of the threshold scheme fully described?  
1681 (b) On what *executing platforms* can the scheme be implemented?  
1682 (c) What are the ~~operational costs and properties of setup and maintenance?~~ What  
1683 ~~are the~~ node-rejuvenation mechanisms (e.g., resharing or ~~node~~-replacement)?  
1684 (d) What are the operational costs and properties of setup and maintenance?  
1685 (e) How are nearby components assumed separate/independent vs. interfering?

R93: A6

1686 2. Applicability of scheme

- 1687 (a) How *efficient/performant* are the operations as a function of threshold parameters?  
1688  
1689 (b) Is the scheme applicable to *NIST-approved* cryptographic primitives?  
1690 (c) Do *base primitives* (e.g., oblivious transfer) require independent standardization?  
1691 (d) Is the *system model* applicable to known and relevant application contexts?

1692 3. Implementations

- 1693 (a) Should the standard take into account feasibility and interoperability on different  
1694 platforms, e.g., hardware or operating systems?  
1695 (b) Should the standard define common APIs for client-side functions?  
1696 (c) What degree of automated protocol validation should be targeted for proposed  
1697 standards?

R94: E9

R95: E10

R96: E11

1698 4. Implementation vs. security

- 1699 (a) How is *diversity* of nodes related to known attack vectors?  
1700 (b) ~~Is the implementation complexity likely to lead to~~ What threshold aspects can  
1701 lead to new implementation bugs or misconfiguration? ~~What~~  
1702 (c) What trusted setup and/assumptions are required (e.g., dealer, special compo-  
1703 nents)?  
1704 (d) How brittle is the scheme (likely to break under small environmental variations)?  
1705  
1706 (e) What faults can be detected and reversed, while identifying the culprit node(s)?

R97: I34

R98: N8

R99: J13

1707 5. Security

- 1708 (a) ~~What threshold properties relate to resistance against side-channel attacks and~~  
1709 ~~how?~~ What threshold properties relate to resistance against side-channel attacks and how?  
1710  
1711 (b) Are there identified *security tradeoffs* across attack types and configurations?  
1712 (c) ~~Is the security assessment supported by a security proof?~~ How does the *reliability*  
1713 compare against that of a conventional implementation?  
1714 (d) ~~How brittle is the scheme (likely to break under small variations in the environment)?~~  
1715 What features of graceful degradation exist against conceivable failures?

- 1716 (e) Does the security proof ensure composability useful for conceived deployments? R100: G7, I38, J8, J9,  
1717 J10,G5, I6, J7, M3, M4
- 1718 (f) Can real attacks thwart the trusted setup assumed in the proof of security? R101: G7, I38, J8, J9,  
1719 (g) To which degree has/have the proof(s) of security been formally verified? J10, E18  
R102: G7, I38, J8, J9,  
J10
- 1720 6. Validation
- 1721 (a) Do the *security assertions* match / fit into the FIPS 140-2 framework?
- 1722 (b) How *testable* is the scheme (can security assertions be tested and validated)?
- 1723 (c) Is there a proposed *automated validation* mechanism?
- 1724 7. Licensing
- 1725 (a) What are the *intellectual property* implications and the *licensing* conditions?
- 1726 8. New standards development
- 1727 Depending on the adopted approach to developing the new standards, there may be  
1728 submissions of candidate threshold schemes for evaluation. If such an approach is  
1729 adopted then potential criteria for quality submissions might include the following:
- 1730 (a) Are *working implementations* available? R103: E6
- 1731 (b) Is *interoperability* between two or more different implementations demonstrated? R104: E7
- 1732
- 1733 (c) Are high-level *use-cases/applications* (e.g., *signing, decryption, etc.*) feasible? R105: E12
- 1734 We need to develop an objective criteria set to support a call for and a selection of  
1735 schemes for standardization. An actual criteria guideline would elaborate further on each  
1736 of the above questions, or variations thereof, and possibly others. The development of such  
1737 criteria would benefit from collaborative public feedback from the cryptography research  
1738 community, as ~~we~~lls-well as from stakeholders in the government and industry.
- 1739 In addition, there may exist pertinent ~~standardization meta-questions~~questions about  
1740 what and how to standardize. What flexibility of parametrization should a standard allow? R106: N9
- 1741 Should there be distinct standardization profiles to separately focus on distinct attribute  
1742 instantiations, e.g., single-device vs. multi-party platform, side-channel attack vs. intrusion  
1743 per node? Next, we elaborate a bit further on two additional aspects.

## 1744 **Standardization at what granularity level?**

### 1745 7.2 Standardization at what granularity level?

1746 Current industry guidelines for best practices in cybersecurity [Ver18] recommend active  
1747 patching of vulnerable components. If in a validated multi-party threshold scheme a node  
1748 is found to have a serious vulnerability, the node may need to be patched. This would  
1749 not be a problem if the scheme tolerates the full compromise of at least one node, and/or  
1750 if it can replace it with another type of (validated) component. In that case, the overall  
1751 system continues to operate smoothly during the patching and revalidation of the vulnerable

1752 component. Thus, when considering the standardization of a particular threshold scheme,  
1753 there may be value in validating implementations with diverse platforms/implementations  
1754 for the individual nodes. This example suggests a question about the standardization criteria:  
1755 what levels of granularity/modularity should be considered for standardization?

1756 While it may be useful to standardize modular components, such development is not  
1757 on its own sufficient to achieve good standards for threshold schemes for cryptographic  
1758 primitives. There are difficulties associated with secure composition of secure components  
1759 into a protocol (e.g., into a threshold scheme). Composability is indeed a recurring subject  
1760 in the development of secure multi-party computation protocols. It is thus an open question  
1761 how one should or should not create standards for combining modular components.

**Standardization opportunities.**  
R108: G5, 16, 37;  
M3, M4

1762 Another consideration on what and how to standardize pertains to the potentially large  
1763 set of available solutions. On the one hand, SMPC provides general techniques that can  
1764 use common building blocks to enable thresholdizing any cryptographic primitive. On  
1765 the other hand, there are also specialized (ad-hoc) solutions with techniques tailored to  
1766 specific primitives and their applications. How should we handle the two types of solutions  
1767 in the standardization process, both of which are likely to improve over time? Some  
1768 benchmarking may be helpful for navigating the pool of possibilities and making objective  
1769 comparisons between ad-hoc and generic solutions. It is important that this process for  
1770 characterizing such solutions is conducted in a way that invites and encourages participation  
1771 from all stakeholders.

R109: G6, J3,  
J4, J5

### 1772 **7.3 Standardization opportunities**

1773 At a basic level, secret-sharing schemes can be used to split a secret key while its use is  
1774 not required, ensuring that a threshold number  $f + 1$  of shares is needed to reconstruct  
1775 the key. However, this by itself does not enable cryptographic primitives to use the key  
1776 shares instead of the recombined key. Secret sharing alone might also not enable threshold  
1777 properties for other purposes, such as preventing a corruption of the intended output. The  
1778 above limitations can be addressed with threshold cryptography (in the broad sense of  
1779 encompassing threshold schemes for a secure implementation of cryptographic primitives).  
1780 For example, the computation may be performed on shares of the key, without the need  
1781 to ever recombine the original key, and enabling threshold security for other properties,  
1782 including integrity and availability. What then should be standardized, within the realm of  
1783 threshold cryptography?

R110: N8

1784 Threshold schemes have wide applicability, in the sense that there are general techniques  
1785 to convert a conventional implementation into a threshold version thereof. One can thus  
1786 ask: for which conventional cryptographic schemes should one consider standardization of  
1787 a threshold version? On one hand, there is a clear interest in enabling threshold versions

1788 of NIST-approved cryptographic primitives. On the other hand, the consideration of stan-  
1789 dardization of threshold schemes is in itself an opportunity to review suitability for new  
1790 standards. In this line, we also wonder how the standardization of threshold schemes might  
1791 also benefit other ongoing NIST efforts of standardization. For example, could elements  
1792 from the lightweight [MBTM17] and post-quantum cryptography (PQC) [NIS17] projects at  
1793 NIST be useful for threshold cryptography? Could the schemes considered by those projects  
1794 also be looked at in the perspective of possible threshold variants? ~~We~~ More research is  
1795 needed. That is why we do not intend here to show any preference about concrete cases, but  
1796 simply to raise the point for consideration. We believe that a better clarification may arise  
1797 from a constructive interaction with the research community and other stakeholders.

R111: E31,E32

## 1798 **8 Conclusions**

1799 Conventional cryptographic implementations have a single point of failure when the secret  
1800 key is stored in one location, or when a localized fault breaks integrity of the output or  
1801 availability of a cryptographic operation. ~~Threshold techniques can mitigate these failure~~  
1802 ~~modes.~~

1803 ~~For example, secret-sharing schemes can be used to split a secret key while its use is~~  
1804 ~~not required, ensuring that a threshold number  $f + 1$  of shares is needed to reconstruct the~~  
1805 ~~key. However, this by itself does not enable cryptographic primitives to use the key shares~~  
1806 ~~instead of the recombined key. In threshold cryptography the computation is performed on~~  
1807 ~~shares of the key, without the need to recombine the original key. A simple secret sharing~~  
1808 ~~might also not enable threshold properties for other goals, such as preventing a corruption~~  
1809 ~~of the intended output. Threshold cryptography may enable operation modes with threshold~~  
1810 ~~security for other properties, including integrity.~~

1811 ~~Generally speaking, we use “threshold cryptography” in the broad sense of encompassing~~  
1812 ~~threshold. These failure modes can be mitigated by using threshold schemes for a secure~~  
1813 implementation of cryptographic primitives. This includes schemes related to secure multi-  
1814 party computation and intrusion-tolerant distributed systems. Usually, a threshold property  
1815 is expressed as an  $f$ -out-of- $n$  tolerance to compromise, where the compromise of up to  $f$   
1816 nodes does not break some security property. For example, when up to  $f$  parties possess  
1817 no information about a secret, security against a wide range of side-channel attacks can be  
1818 achieved under some reasonable assumptions about the distributions of side-channel infor-  
1819 mation. Furthermore, a threshold scheme may even provide resistance against side-channel  
1820 attacks that collect information correlated across all nodes (beyond the threshold). This  
1821 is because, in some models, a threshold design may complicate the exploitation of noisy  
1822 side-channel information.

R112: See R110

1823 Threshold schemes can be efficient. For example, we described how a simple threshold  
1824 RSA signature scheme based on a  ~~$n$ -out-~~ out-of- $n$  secret-sharing has a complexity that  
1825 increases only linearly with the number of shares, and whose computation is parallelized

1826 across several nodes. In such case, the simplicity of the method is based on a mathematical  
1827 ~~structure property~~ (a homomorphism) ~~present in of~~ the underlying structure of the original  
1828 scheme. In contrast, schemes for other cryptographic primitives, such as some block ciphers,  
1829 may require significant computational overhead compared to their conventional counterparts.  
1830 Still, even in those cases the threshold schemes may be practical and justifiable, depending  
1831 on the intended security assurances and the application context.

1832 The discussion in the preceding sections highlighted nuanced security assertions that can  
1833 be obtained about threshold cryptographic schemes. The security of such schemes has to be  
1834 seen through the prism of a security model and possibly considering several system models  
1835 of implementation. For example, there may be differences between active or passive attacks.  
1836 To help navigate the landscape of possible schemes, this report enumerated characterizing  
1837 features and their possible effects. For example: there are potential benefits of rerandomizing  
1838 the shares of the secret key; properties can be different between multi-device vs. single-  
1839 device platforms; some security properties are different depending on the communication  
1840 platform with the environment and between components.

1841 An understanding of a variety of instantiations of characterizing features is necessary  
1842 for the development of objective criteria for selecting candidates for standardization. For the  
1843 purpose of standardization and validation, the combination of characterizing features and  
1844 attack models should be translated into security assertions. The way these can fit into FIPS  
1845 140-2 and/or require complementary standardization is a matter for discussion.

1846 We have looked at numerous factors that influence the type of security assessment  
1847 that can be made about threshold cryptographic schemes. Clearly, threshold cryptography  
1848 has potential to improve the security of the implementation of cryptographic primitives,  
1849 provided it is carefully used. There is a clear interest in enabling threshold schemes for  
1850 already standardized cryptographic primitives. The standardization effort may also constitute  
1851 an opportunity to consider the case for standardizing new primitives. There are long-standing  
1852 research results, and the research community is still active in the area.

1853 We intend this report to initiate a discussion on the standardization of threshold cryp-  
1854 tographic schemes. We can envision some of the challenges ahead. The most immediate  
1855 seems to be the development of criteria for and selection of proposals. This document did  
1856 not put forth such criteria, but motivated the need for one and developed some basis for it.

1857 Once criteria are in place, the selection and standardization of concrete schemes should  
1858 include an integration with validation methodologies. How then to express security asser-  
1859 tions that may fit within FIPS 140-2 or fit well as a complement thereof? What security  
1860 and implementation profiles should be devised? When ~~tacking~~ tackling these challenges,  
1861 positive synergies may result from engaging with and incorporating feedback from the  
1862 research community and other stakeholders.

1863 **References**

- 1864 [ABF<sup>+</sup>03] C. Aumüller, P. Bier, W. Fischer, P. Hofreiter, and J.-P. Seifert. *Fault Attacks*  
 1865 *on RSA with CRT: Concrete Results and Practical Countermeasures*. In  
 1866 B. S. Kaliski, Ç. K. Koç, and C. Paar (eds.), *Cryptographic Hardware and*  
 1867 *Embedded Systems — CHES 2002*, vol. 2523 of LNCS, pages 260–275.  
 1868 Springer Berlin Heidelberg, 2003. DOI:[10.1007/3-540-36400-5\\_20](https://doi.org/10.1007/3-540-36400-5_20).
- 1869 [AK96] R. Anderson and M. Kuhn. *Tamper Resistance: A Cautionary Note*. Proc.  
 1870 2nd USENIX Workshop on Electronic Commerce (WOEC'96), 2:1–11, 1996.
- 1871 [AMD12] AMD Corporation. *AMD has you covered*. [https://www.amd.com/Documents/](https://www.amd.com/Documents/Security_021.pdf)  
 1872 [Security\\_021.pdf](https://www.amd.com/Documents/Security_021.pdf), 2012.
- 1873 [AMGC85] B. Awerbuch, S. Micali, S. Goldwasser, and B. Chor. *Verifiable secret sharing*  
 1874 *and achieving simultaneity in the presence of faults*. In Proc. 26th Annual  
 1875 Symposium on Foundations of Computer Science, SFCS '85, pages 383–395.  
 1876 IEEE Computer Society, 1985. DOI:[10.1109/SFCS.1985.64](https://doi.org/10.1109/SFCS.1985.64).
- 1877 [AMN01] M. Abdalla, S. Miner, and C. Namprempe. *Forward-Secure Threshold*  
 1878 *Signature Schemes*. In D. Naccache (ed.), *Topics in Cryptology — CT-RSA*  
 1879 *2001*, pages 441–456. Springer Berlin Heidelberg, 2001. DOI:[10.1007/3-540-](https://doi.org/10.1007/3-540-45353-9_32)  
 1880 [45353-9\\_32](https://doi.org/10.1007/3-540-45353-9_32).
- 1881 [And02] R. Anderson. *Two remarks on public key cryptology*. Technical report,  
 1882 University of Cambridge, Computer Laboratory, 2002.
- 1883 [ARM18] ARM Corporation. *TrustZone*. [https://www.arm.com/products/](https://www.arm.com/products/security-on-arm/trustzone)  
 1884 [security-on-arm/trustzone](https://www.arm.com/products/security-on-arm/trustzone), 2018.
- 1885 [BB03] D. Brumley and D. Boneh. *Remote Timing Attacks Are Practical*. In Proc.  
 1886 12th Conference on USENIX Security Symposium — SSYM'03, pages 1–13.  
 1887 USENIX Association, 2003.
- 1888 [BB12] L. T. A. N. Brandão and A. N. Bessani. *On the reliability and avail-*  
 1889 *ability of replicated and rejuvenating systems under stealth attacks and*  
 1890 *intrusions*. *Journal of the Brazilian Computer Society*, 18(1):61–80, 2012.  
 1891 DOI:[10.1007/s13173-012-0062-x](https://doi.org/10.1007/s13173-012-0062-x).
- 1892 [BBK17] [K. Bhargavan, B. Blanchet, and N. Kobeissi](https://doi.org/10.1109/SP.2017.26). [Verified Models and Reference](https://doi.org/10.1109/SP.2017.26)  
 1893 [Implementations for the TLS 1.3 Standard Candidate](https://doi.org/10.1109/SP.2017.26). In [2017 IEEE](https://doi.org/10.1109/SP.2017.26)  
 1894 [Symposium on Security and Privacy \(SP\)](https://doi.org/10.1109/SP.2017.26), pages 483–502. IEEE, may 2017.  
 1895 DOI:[10.1109/SP.2017.26](https://doi.org/10.1109/SP.2017.26).
- 1896 [BCF00] E. F. Brickell, G. D. Crescenzo, and Y. Frankel. *Sharing Block Ciphers*. In  
 1897 *Information Security and Privacy – ACISP 2000*, vol. 1841 of LNCS, pages  
 1898 457–470. Springer Berlin Heidelberg, 2000. DOI:[10.1007/10718964\\_37](https://doi.org/10.1007/10718964_37).

- 1899 [BDK17] J. Behl, T. Distler, and R. Kapitza. *Hybrids on Steroids: SGX-Based*  
1900 *High Performance BFT*. In Proc. 12th European Conference on Computer  
1901 Systems, EuroSys '17, pages 222–237, New York, NY, USA, 2017. ACM.  
1902 DOI:10.1145/3064176.3064213.
- 1903 [BDL97] D. Boneh, R. A. DeMillo, and R. J. Lipton. *On the Importance of Checking*  
1904 *Cryptographic Protocols for Faults*. In W. Fumy (ed.), *Advances in*  
1905 *Cryptology — EUROCRYPT '97*, vol. 1233 of LNCS, pages 37–51. Springer  
1906 Berlin Heidelberg, 1997. DOI:10.1007/3-540-69053-0\_4.
- 1907 [Bea92] D. Beaver. *Efficient Multiparty Protocols Using Circuit Randomiza-*  
1908 *tion*. In J. Feigenbaum (ed.), *Advances in Cryptology — CRYPTO '91*,  
1909 vol. 576 of LNCS, pages 420–432. Springer Berlin Heidelberg, 1992.  
1910 DOI:10.1007/3-540-46766-1\_34.  
1911 ~~[BO83]~~
- 1912 [Ben83] M. Ben-Or. *Another Advantage of Free Choice (Extended Abstract):*  
1913 *Completely Asynchronous Agreement Protocols*. In Proc. 2nd Annual ACM  
1914 Symposium on Principles of Distributed Computing, PODC '83, pages 27–30.  
1915 ACM, 1983. DOI:10.1145/800221.806707.
- 1916 [Ber05] D. J. Bernstein. *Cache-timing attacks on AES*. [https://cr.yp.to/antiforgery/  
1917 cachetiming-20050414.pdf](https://cr.yp.to/antiforgery/cachetiming-20050414.pdf), 2005.
- 1918 [Ber06] D. J. Bernstein. *Curve25519: New Diffie-Hellman Speed Records*. In M. Yung,  
1919 Y. Dodis, A. Kiayias, and T. Malkin (eds.), *Public Key Cryptography — PKC*  
1920 2006, vol. 3958 of LNCS, pages 207–228. Springer Berlin Heidelberg, 2006.  
1921 DOI:10.1007/11745853\_14.
- 1922 [BF97] D. Boneh and M. Franklin. *Efficient generation of shared RSA keys*. In B. S.  
1923 Kaliski (ed.), *Advances in Cryptology — CRYPTO '97*, vol. 1294 of LNCS,  
1924 pages 425–439. Springer Berlin Heidelberg, 1997. DOI:10.1007/BFb0052253.
- 1925 [BFM88] M. Blum, P. Feldman, and S. Micali. *Non-interactive Zero-knowledge and Its*  
1926 *Applications*. In Proc. 20th Annual ACM Symposium on Theory of Computing,  
1927 STOC '88, pages 103–112. ACM, 1988. DOI:10.1145/62212.62222.
- 1928 [BGG<sup>+</sup>14] J. Balasch, B. Gierlichs, V. Grosso, O. Reparaz, and F. Standaert. *On the Cost*  
1929 *of Lazy Engineering for Masked Software Implementations*. In *Smart Card*  
1930 *Research and Advanced Applications — CARDIS*, vol. 8968 of LNCS, pages  
1931 64–81. Springer Berlin Heidelberg, 2014. DOI:10.1007/978-3-319-16763-3\_5.
- 1932 [BGW88] M. Ben-Or, S. Goldwasser, and A. Wigderson. *Completeness Theorems*  
1933 *for Non-cryptographic Fault-tolerant Distributed Computation*. In  
1934 *Proceedings of the Twentieth Annual ACM Symposium on Theory of*  
1935 *Computing, STOC '88, pages 1–10, New York, NY, USA, 1988. ACM.*  
1936 DOI:10.1145/62212.62213.

- 1937 [BH98] [D. Boneh and J. Horwitz. \*Generating a product of three primes with an\*](#)  
1938 [unknown factorization. In J. P. Buhler \(ed.\), \*Algorithmic Number Theory,\*](#)  
1939 [LNCS, pages 237–251, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.](#)  
1940 [DOI:10.1007/BFb0054866.](#)
- 1941 [Bla79] G. R. Blakley. *Safeguarding cryptographic keys.* In Proc. International  
1942 Workshop on Managing Requirements Knowledge, vol. 48 of AFIPS 1979,  
1943 pages 313–317, 1979. DOI:[10.1109/AFIPS.1979.98.](#)
- 1944 [BM99] M. Bellare and S. K. Miner. *A Forward-Secure Digital Signature Scheme.*  
1945 In M. Wiener (ed.), *Advances in Cryptology — CRYPTO’ 99*, vol. 1666 of  
1946 LNCS, pages 431–448. Springer Berlin Heidelberg, 1999. DOI:[10.1007/3-](#)  
1947 [540-48405-1\\_28.](#)
- 1948 [BMW<sup>+</sup>18] [J. v. Bulck, M. Minkin, O. Weisse, D. Genkin, B. Kasikci, F. Piessens,](#)  
1949 [M. Silberstein, T. F. Wenisch, Y. Yarom, and R. Strackx. \*Foreshadow:\*](#)  
1950 [\*Extracting the Keys to the Intel SGX Kingdom with Transient Out-of-Order\*](#)  
1951 [\*Execution.\* In 27th USENIX Security Symposium \(USENIX Security 18\),](#)  
1952 [page 991–1008, Baltimore, MD, 2018. USENIX Association.](#)
- 1953 [BN06] M. Bellare and G. Neven. *Multi-signatures in the Plain public-Key Model*  
1954 [and a General Forking Lemma. In Proc. 13th ACM Conference on Computer](#)  
1955 [and Communications Security, CCS ’06, pages 390–399. ACM, 2006.](#)  
1956 [DOI:10.1145/1180405.1180453.](#)
- 1957 [BR07] [M. Bellare and P. Rogaway. \*Robust Computational Secret Sharing and\*](#)  
1958 [a Unified Account of Classical Secret-sharing Goals. In Proceedings](#)  
1959 [of the 14th ACM Conference on Computer and Communications](#)  
1960 [Security, CCS ’07, pages 172–184, New York, NY, USA, 2007. ACM.](#)  
1961 [DOI:10.1145/1315245.1315268.](#)
- 1962 [Bre12] E. Brewer. *CAP Twelve Years Later: How the "Rules" Have Changed.*  
1963 Computer, 45:23–29, 01 2012. DOI:[10.1109/MC.2012.37.](#)
- 1964 [CA78] L. Chen and A. Avizienis. *N-Version Programming: A Fault-Tolerance*  
1965 [Approach to Reliability of Software Operation. Digest FTCS-8: 8th Annual](#)  
1966 [International Conference on Fault Tolerant Computing, pages 3–9, June 1978.](#)
- 1967 [Can01] R. Canetti. *Universally Composable Security: A New Paradigm for*  
1968 [Cryptographic Protocols. In Proc. 42nd IEEE Symposium on Foundations](#)  
1969 [of Computer Science, FOCS ’01, pages 136–145. IEEE Computer Society,](#)  
1970 [2001. DOI:10.1109/SFCS.2001.959888.](#)
- 1971 [CCD88] [D. Chaum, C. Crépeau, and I. Damgard. \*Multiparty Unconditionally Secure\*](#)  
1972 [Protocols. In Proceedings of the Twentieth Annual ACM Symposium on](#)  
1973 [Theory of Computing, STOC ’88, pages 11–19, New York, NY, USA, 1988.](#)  
1974 [ACM. DOI:10.1145/62212.62214.](#)

- 1975 [CHH<sup>+</sup>17] [C. Cremers, M. Horvat, J. Hoyland, S. Scott, and T. van der Merwe.](#)  
1976 [A Comprehensive Symbolic Analysis of TLS 1.3.](#) In [Proceedings of the](#)  
1977 [2017 ACM SIGSAC Conference on Computer and Communications](#)  
1978 [Security, CCS '17, pages 1773–1788, New York, NY, USA, 2017. ACM.](#)  
1979 [DOI:10.1145/3133956.3134063.](#)
- 1980 [CJRR99] S. Chari, C. S. Jutla, J. R. Rao, and P. Rohatgi. *Towards Sound Approaches*  
1981 *to Counteract Power-Analysis Attacks.* In M. Wiener (ed.), *Advances in*  
1982 *Cryptology — CRYPTO' 99*, vol. 1666 of LNCS, pages 398–412. Springer  
1983 Berlin Heidelberg, 1999. DOI:[10.1007/3-540-48405-1\\_26](#).
- 1984 [CL02] M. Castro and B. Liskov. *Practical Byzantine Fault Tolerance and Proactive*  
1985 *Recovery.* *ACM Trans. Comput. Syst.*, 20(4):398–461, November 2002.  
1986 DOI:[10.1145/571637.571640](#).
- 1987 [Com17] Common Criteria. *Common Criteria for Information Technology Security*  
1988 *Evaluation*, April 2017. <https://www.commoncriteriaportal.org/>.
- 1989 [CvH91] D. Chaum and E. van Heyst. *Group Signatures.* In D. W. Davies (ed.),  
1990 *Advances in Cryptology — EUROCRYPT '91*, vol. 547 of LNCS, pages  
1991 257–265. Springer Berlin Heidelberg, 1991. DOI:[10.1007/3-540-46416-6\\_22](#).
- 1992 [DDF14] A. Duc, S. Dziembowski, and S. Faust. *Unifying Leakage Models: From Prob-*  
1993 *ing Attacks to Noisy Leakage.* In *Advances in Cryptology — EUROCRYPT*  
1994 *2014*, vol. 8441 of LNCS, pages 423–440. Springer Berlin Heidelberg, 2014.  
1995 DOI:[10.1007/978-3-642-55220-5\\_24](#).
- 1996 [DDN03] D. Dolev, C. Dwork, and M. Naor. *Nonmalleable Cryptography.* *SIAM*  
1997 *Review*, 45(4):727–784, 2003. DOI:[10.1137/S0036144503429856](#).
- 1998 [DDS87] D. Dolev, C. Dwork, and L. Stockmeyer. *On the Minimal Synchronism*  
1999 *Needed for Distributed Consensus.* *J. ACM*, 34(1):77–97, January 1987.  
2000 DOI:[10.1145/7531.7533](#).
- 2001 [DF90] Y. Desmedt and Y. Frankel. *Threshold cryptosystems.* In G. Brassard (ed.),  
2002 *Advances in Cryptology — CRYPTO' 89 Proceedings*, vol. 435 of LNCS,  
2003 pages 307–315. Springer New York, 1990. DOI:[10.1007/0-387-34805-0\\_28](#).
- 2004 [DJ97] [Y. Desmedt and S. Jajodia.](#) [Redistributing Secret Shares to New Access](#)  
2005 [Structures and Its Applications.](#) Technical Report ISSE-TR-97-01, George  
2006 [Mason University, July 1997.](#)
- 2007 [DLFK<sup>+</sup>17] [A. Delignat-Lavaud, C. Fournet, M. Kohlweiss, J. Protzenko, A. Rastogi,](#)  
2008 [N. Swamy, S. Zanella-Béguelin, K. Bhargavan, J. Pan, and J. K.](#)  
2009 [Zinzindohoué.](#) *Implementing and proving the TLS 1.3 record layer.* In  
2010 *2017 IEEE Symposium on Security and Privacy (SP)*, pages 463–482. IEEE,  
2011 *May 2017.* DOI:[10.1109/SP.2017.58](#).

- 2012 [DLK<sup>+</sup>14] Z. Durumeric, F. Li, J. Kasten, J. Amann, J. Beekman, M. Payer, N. Weaver,  
2013 D. Adrian, V. Paxson, M. Bailey, and J. A. Halderman. *The Matter of*  
2014 *Heartbleed*. In Proc. 2014 Conference on Internet Measurement Con-  
2015 ference, IMC '14, pages 475–488, New York, NY, USA, 2014. ACM.  
2016 DOI:[10.1145/2663716.2663755](https://doi.org/10.1145/2663716.2663755).
- 2017 [Dou02] J. R. Douceur. *The Sybil Attack*. In P. Druschel, F. Kaashoek, and A. Rowstron  
2018 (eds.), *Peer-to-Peer Systems*, pages 251–260. Springer Berlin Heidelberg,  
2019 2002. DOI:[10.1007/3-540-45748-8\\_24](https://doi.org/10.1007/3-540-45748-8_24).
- 2020 [DSDFY94] A. De Santis, Y. Desmedt, Y. Frankel, and M. Yung. *How to Share a Function*  
2021 *Securely*. In Proc. 26th Annual ACM Symposium on Theory of Computing,  
2022 STOC '94, pages 522–533. ACM, 1994. DOI:[10.1145/195058.195405](https://doi.org/10.1145/195058.195405).
- 2023 [Fel87] P. Feldman. *A Practical Scheme for Non-interactive Verifiable Secret*  
2024 *Sharing*. In Proc. 28th Annual Symposium on Foundations of Computer  
2025 Science, SFCS '87, pages 427–438. IEEE Computer Society, 1987.  
2026 DOI:[10.1109/SFCS.1987.4](https://doi.org/10.1109/SFCS.1987.4).
- 2027 [FHHW03] [M. Fitzi, M. Hirt, T. Holenstein, and J. Wullschlegler. \*Two-Threshold\*](https://doi.org/10.1007/3-540-39200-9_4)  
2028 [\*Broadcast and Detectable Multi-party Computation\*. In E. Biham \(ed.\),](https://doi.org/10.1007/3-540-39200-9_4)  
2029 [\*Advances in Cryptology — EUROCRYPT 2003\*, vol. 2656 of LNCS, pages](https://doi.org/10.1007/3-540-39200-9_4)  
2030 [51–67, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.](https://doi.org/10.1007/3-540-39200-9_4) DOI:[10.1007/3-](https://doi.org/10.1007/3-540-39200-9_4)  
2031 [540-39200-9\\_4](https://doi.org/10.1007/3-540-39200-9_4).
- 2032 [FLP85] M. J. Fischer, N. A. Lynch, and M. S. Paterson. *Impossibility of Distributed*  
2033 *Consensus with One Faulty Process*. *J. ACM*, 32(2):374–382, April 1985.  
2034 DOI:[10.1145/3149.214121](https://doi.org/10.1145/3149.214121).
- 2035 [FNP04] M. J. Freedman, K. Nissim, and B. Pinkas. *Efficient private matching and*  
2036 *set intersection*. In International conference on the theory and applications  
2037 of cryptographic techniques, pages 1–19. Springer, 2004.
- 2038 [Fra90] Y. Frankel. *A Practical Protocol for Large Group Oriented Networks*.  
2039 In J.-J. Quisquater and J. Vandewalle (eds.), *Advances in Cryptology —*  
2040 *EUROCRYPT '89*, LNCS, pages 56–61, Berlin, Heidelberg, 1990. Springer  
2041 Berlin Heidelberg. DOI:[10.1007/3-540-46885-4\\_8](https://doi.org/10.1007/3-540-46885-4_8).
- 2042 [GJKR99] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. *Secure Distributed*  
2043 *Key Generation for Discrete-Log Based Cryptosystems*. In J. Stern (ed.),  
2044 *Advances in Cryptology — EUROCRYPT '99*, vol. 1592 of LNCS,  
2045 pages 295–310, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg.  
2046 DOI:[10.1007/3-540-48910-X\\_21](https://doi.org/10.1007/3-540-48910-X_21).
- 2047 [GMR85] S. Goldwasser, S. Micali, and C. Rackoff. *The Knowledge Complexity*  
2048 *of Interactive Proof-systems*. In Proc. 17th Annual ACM Symposium  
2049 on Theory of Computing, STOC '85, pages 291–304. ACM, 1985.

- 2050 DOI:[10.1145/22145.22178](https://doi.org/10.1145/22145.22178).
- 2051 [GMW87] O. Goldreich, S. Micali, and A. Wigderson. *How to Play ANY Mental Game*  
2052 *or A Completeness Theorem for Protocols with Honest Majority*. In Proc.  
2053 19th Annual ACM Symposium on Theory of Computing, STOC '87, pages  
2054 218–229. ACM, 1987. DOI:[10.1145/28395.28420](https://doi.org/10.1145/28395.28420).
- 2055 [GRJK00] R. Gennaro, T. Rabin, S. Jarecki, and H. Krawczyk. *Robust and Efficient*  
2056 *Sharing of RSA Functions*. Journal of Cryptology, 13(2):273–300, Mar 2000.  
2057 DOI:[10.1007/s001459910011](https://doi.org/10.1007/s001459910011).
- 2058 [Gro16] C. T. Group. *NIST Cryptographic Standards and Guidelines Development*  
2059 *Process*. NISTIR 7977, March 2016. DOI:[10.6028/NIST.IR.7977](https://doi.org/10.6028/NIST.IR.7977).
- 2060 [Gue09] S. Gueron. *Intel's New AES Instructions for Enhanced Performance and*  
2061 *Security*. In O. Dunkelman (ed.), Fast Software Encryption, 16th International  
2062 Workshop, FSE 2009, vol. 5665 of LNCS, pages 51–66. Springer, 2009.  
2063 DOI:[10.1007/978-3-642-03317-9\\_4](https://doi.org/10.1007/978-3-642-03317-9_4).
- 2064 [HJKY95] A. Herzberg, S. Jarecki, H. Krawczyk, and M. Yung. *Proactive Secret Sharing*  
2065 *Or: How to Cope With Perpetual Leakage*. In D. Coppersmith (ed.), Advances  
2066 in Cryptology — CRYPTO' 95, pages 339–352. Springer Berlin Heidelberg,  
2067 1995. DOI:[10.1007/3-540-44750-4\\_27](https://doi.org/10.1007/3-540-44750-4_27).
- 2068 [HM00] M. Hirt and U. Maurer. *Player Simulation and General Adversary Structures*  
2069 *in Perfect Multiparty Computation*. Journal of Cryptology, 13(1):31–60, Jan  
2070 2000. DOI:[10.1007/s001459910003](https://doi.org/10.1007/s001459910003).
- 2071 [HSH<sup>+</sup>09] J. A. Halderman, S. D. Schoen, N. Heninger, W. Clarkson, W. Paul, J. A.  
2072 Calandrino, A. J. Feldman, J. Appelbaum, and E. W. Felten. *Lest We*  
2073 *Remember: Cold-boot Attacks on Encryption Keys*. Commun. ACM,  
2074 52(5):91–98, May 2009. DOI:[10.1145/1506409.1506429](https://doi.org/10.1145/1506409.1506429).
- 2075 [IN83] K. Itakura and K. Nakamura. *A public-key cryptosystem suitable for digital*  
2076 *multisignatures*. In NEC J. Res. Dev. 71, pages 1–8, Oct. 1983.
- 2077 [Int18] Intel Corporation. *Software Guard Extention (SGX)*. <https://software.intel.com/en-us/sgx>, 2018.  
2078
- 2079 [ISN89] M. Ito, A. Saito, and T. Nishizeki. *Secret sharing scheme realizing general ac-*  
2080 *cess structure*. Electronics and Communications in Japan (Part III: Fundamen-  
2081 tal Electronic Science), 72(9):56–64, 1989. DOI:[10.1002/ecjc.4430720906](https://doi.org/10.1002/ecjc.4430720906).
- 2082 [ISO12] ISO. *ISO/IEC 19790:2012, Information technology – Security*  
2083 *techniques – Security requirements for cryptographic modules*.  
2084 <https://www.iso.org/standard/52906.html>, August 2012.
- 2085 [ISO16] ISO. *ISO/IEC 19592-1:2016, Information technology – Security techniques*  
2086 *– Secret sharing – Part 1: General*. <https://www.iso.org/standard/65422.html>,

- 2087                   2016.
- 2088       [ISO17] ISO. *ISO/IEC 19592-2:2017, Information technology – Security*  
2089 *techniques – Secret sharing – Part 2: Fundamental mechanisms.*  
2090 <https://www.iso.org/standard/65425.html>, 2017.
- 2091       [ISW03] Y. Ishai, A. Sahai, and D. A. Wagner. *Private Circuits: Securing Hardware*  
2092 *against Probing Attacks.* In *Advances in Cryptology — CRYPTO 2003*,  
2093 vol. 2729 of LNCS, pages 463–481. Springer Berlin Heidelberg, 2003.  
2094 DOI:[10.1007/978-3-540-45146-4\\_27](https://doi.org/10.1007/978-3-540-45146-4_27).
- 2095       [JB99] A. Juels and J. Brainard. *Client puzzles: A Cryptographic countermeasure*  
2096 *against connection depletion attacks.* In *Network and distributed system secu-*  
2097 *rity symposium — NDSS’99*, vol. 99, pages 151–168. Internet Society, 1999.
- 2098       [KF95] N. Kolettis and N. D. Fulton. *Software Rejuvenation: Analysis, Module and Ap-*  
2099 *plications.* In *Proc. 25th International Symposium on Fault-Tolerant Comput-*  
2100 *ing, FTCS ’95*, pages 381–390. IEEE, 1995. DOI:[10.1109/FTCS.1995.466961](https://doi.org/10.1109/FTCS.1995.466961).
- 2101       [KGG<sup>+</sup>18] P. Kocher, D. Genkin, D. Gruss, W. Haas, M. Hamburg, M. Lipp, S. Mangard,  
2102 T. Prescher, M. Schwarz, and Y. Yarom. *Spectre Attacks: Exploiting*  
2103 *Speculative Execution.* [ArXiv e-prints arXiv:1801.01203](https://arxiv.org/abs/1801.01203), January 2018.
- 2104       [Kis13] R. Kissel. *Glossary of Key Information Security Terms.* NISTIR 7298  
2105 Revision 2, May 2013. DOI:[10.6028/NIST.IR.7298r2](https://doi.org/10.6028/NIST.IR.7298r2).
- 2106       [KK07] I. Koren and C. M. Krishna. *Fault-Tolerant Systems.* Morgan Kaufmann  
2107 Publishers Inc., 1st edition, 2007.
- 2108       [Koc96] P. C. Kocher. *Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS,*  
2109 *and Other Systems.* In N. Kobitz (ed.), *Advances in Cryptology — CRYPTO*  
2110 *’96*, vol. 1109 of LNCS, pages 104–113. Springer Berlin Heidelberg, 1996.  
2111 DOI:[10.1007/3-540-68697-5\\_9](https://doi.org/10.1007/3-540-68697-5_9).
- 2112       [KPVV16] T. Kaufmann, H. Pelletier, S. Vaudenay, and K. Villegas. *When Constant-Time*  
2113 *Source Yields Variable-Time Binary: Exploiting Curve25519-donna Built*  
2114 *with MSVC 2015.* In *Cryptology and Network Security — CANS 2016*,  
2115 vol. 10052 of LNCS, pages 573–582. Springer Berlin Heidelberg, 2016.  
2116 DOI:[10.1007/978-3-319-48965-0\\_36](https://doi.org/10.1007/978-3-319-48965-0_36).
- 2117       [Kra94] H. Krawczyk. *Secret Sharing Made Short.* In D. R. Stinson (ed.), *Advances*  
2118 *in Cryptology — CRYPTO ’93*, vol. 573 of LNCS, pages 136–146. Springer  
2119 Berlin Heidelberg, 1994. DOI:[10.1007/3-540-48329-2\\_12](https://doi.org/10.1007/3-540-48329-2_12).
- 2120       [Lam06] L. Lamport. *Lower bounds for asynchronous consensus.* *Distributed*  
2121 *Computing*, 19(2):104–125, Oct 2006. DOI:[10.1007/s00446-006-0155-x](https://doi.org/10.1007/s00446-006-0155-x).
- 2122       [LS04] B. Littlewood and L. Strigini. *Redundancy and Diversity in Security.*  
2123 In P. Samarati, P. Ryan, D. Gollmann, and R. Molva (eds.), *Computer*

- 2124 Security – ESORICS 2004, pages 423–438. Springer Berlin Heidelberg, 2004.  
2125 DOI:[10.1007/978-3-540-30108-0\\_26](https://doi.org/10.1007/978-3-540-30108-0_26).
- 2126 [LSG<sup>+</sup>18] M. Lipp, M. Schwarz, D. Gruss, T. Prescher, W. Haas, S. Mangard,  
2127 P. Kocher, D. Genkin, Y. Yarom, and M. Hamburg. *Meltdown*. [ArXiv  
2128 e-printsarXiv:1801.01207](https://arxiv.org/abs/1801.01207), January 2018.
- 2129 [LSP82] L. Lamport, R. Shostak, and M. Pease. *The Byzantine Generals Problem*.  
2130 ACM Transactions on Programming Languages and Systems, 4(3):382–401,  
2131 July 1982. DOI:[10.1145/357172.357176](https://doi.org/10.1145/357172.357176).
- 2132 [Lyn89] N. Lynch. *A Hundred Impossibility Proofs for Distributed Computing*. In  
2133 Proc. 8th Annual ACM Symposium on Principles of Distributed Computing,  
2134 PODC '89, pages 1–28. ACM, 1989. DOI:[10.1145/72981.72982](https://doi.org/10.1145/72981.72982).
- 2135 [MBTM17] K. A. McKay, L. Bassham, M. S. Turan, and N. Mouha. *NISTIR 8114 —  
2136 Report on Lightweight Cryptography*, 2017. DOI:[10.6028/NIST.IR.8114](https://doi.org/10.6028/NIST.IR.8114).
- 2137 [MOR01] S. Micali, K. Ohta, and L. Reyzin. *Accountable-subgroup Multisigna-  
2138 tures: Extended Abstract*. In Proc. 8th ACM Conference on Computer  
2139 and Communications Security, CCS '01, pages 245–254. ACM, 2001.  
2140 DOI:[10.1145/501983.502017](https://doi.org/10.1145/501983.502017).
- 2141 [Mor11] T. Morris. *Trusted Platform Module*. In H. C. A. van Tilborg and S. Jajodia  
2142 (eds.), *Encyclopedia of Cryptography and Security*, 2nd Ed., pages 1332–1335.  
2143 Springer, 2011. DOI:[10.1007/978-1-4419-5906-5\\_796](https://doi.org/10.1007/978-1-4419-5906-5_796).
- 2144 [NIS01] NIST. *Security Requirements for Cryptographic Modules, Federal Information  
2145 Processing Standard (FIPS) 140-2*, 2001. DOI:[10.6028/NIST.FIPS.140-2](https://doi.org/10.6028/NIST.FIPS.140-2).
- 2146 [NIS17] NIST. *Post-quantum Cryptography Project*. [https://csrc.nist.gov/projects/  
2147 post-quantum-cryptography](https://csrc.nist.gov/projects/post-quantum-cryptography), 2017.
- 2148 [NIS18a] NIST. *Automated Cryptographic Validation Protocol*. [https:  
2149 //github.com/usnistgov/ACVP](https://github.com/usnistgov/ACVP), 2018.
- 2150 [NIS18b] NIST. *Automated Cryptographic Validation Testing*. [https:  
2151 //csrc.nist.gov/projects/acvt/](https://csrc.nist.gov/projects/acvt/), 2018.
- 2152 [NIS18c] NIST. *Cryptographic Module Validation Program*. [https://csrc.nist.gov/  
2153 projects/cryptographic-module-validation-program](https://csrc.nist.gov/projects/cryptographic-module-validation-program), 2018.
- 2154 [NRR06] S. Nikova, C. Rechberger, and V. Rijmen. *Threshold Implementations  
2155 Against Side-Channel Attacks and Glitches*. In P. Ning, S. Qing, and  
2156 N. Li (eds.), *Information and Communications Security — ICICS 2006*,  
2157 vol. 4307 of LNCS, pages 529–545. Springer Berlin Heidelberg, 2006.  
2158 DOI:[10.1007/11935308\\_38](https://doi.org/10.1007/11935308_38).
- 2159 [NVD14] NVD. *National Vulnerability Database — CVE-2014-0160*.

- 2160 <https://nvd.nist.gov/vuln/detail/CVE-2014-0160>, 2014.
- 2161 [NVD18a] NVD. *National Vulnerability Database — CVE-2017-5715*.  
2162 <https://nvd.nist.gov/vuln/detail/CVE-2017-5715>, 2018.
- 2163 [NVD18b] NVD. *National Vulnerability Database — CVE-2017-5753*.  
2164 <https://nvd.nist.gov/vuln/detail/CVE-2017-5753>, 2018.
- 2165 [NVD18c] NVD. *National Vulnerability Database — CVE-2017-5754*.  
2166 <https://nvd.nist.gov/vuln/detail/CVE-2017-5754>, 2018.
- 2167 [OY91] R. Ostrovsky and M. Yung. *How to Withstand Mobile Virus Attacks*  
2168 *(Extended Abstract)*. In Proc. 10th Annual ACM Symposium on Prin-  
2169 ciples of Distributed Computing, PODC '91, pages 51–59. ACM, 1991.  
2170 DOI:[10.1145/112600.112605](https://doi.org/10.1145/112600.112605).
- 2171 [Ped91] T. P. Pedersen. *A Threshold Cryptosystem without a Trusted Party*. In  
2172 D. W. Davies (ed.), *Advances in Cryptology — EUROCRYPT '91*,  
2173 vol. 547 of LNCS, pages 522–526. Springer Berlin Heidelberg, 1991.  
2174 DOI:[10.1007/3-540-46416-6\\_47](https://doi.org/10.1007/3-540-46416-6_47).
- 2175 [Ped92] T. P. Pedersen. *Non-Interactive and Information-Theoretic Secure Verifiable*  
2176 *Secret Sharing*. In J. Feigenbaum (ed.), *Advances in Cryptology — CRYPTO*  
2177 *'91*, vol. 576 of LNCS, pages 129–140. Springer Berlin Heidelberg, 1992.  
2178 DOI:[10.1007/3-540-46766-1\\_9](https://doi.org/10.1007/3-540-46766-1_9).
- 2179 [Por18] T. Pornin. *BearSSL — Constant-Time Mul*. <https://bearssl.org/ctmul.html>,  
2180 2018.
- 2181 [PSL80] M. Pease, R. Shostak, and L. Lamport. *Reaching Agreement in the Presence*  
2182 *of Faults*. J. ACM, 27(2):228–234, April 1980. DOI:[10.1145/322186.322188](https://doi.org/10.1145/322186.322188).
- 2183 [PW92] B. Pfitzmann and M. Waidner. *Unconditional Byzantine agreement for any*  
2184 *number of faulty processors*. In A. Finkel and M. Jantzen (eds.), *STACS 92*,  
2185 pages 337–350. Springer Berlin Heidelberg, 1992. DOI:[10.1007/3-540-55210-](https://doi.org/10.1007/3-540-55210-3_195)  
2186 [3\\_195](https://doi.org/10.1007/3-540-55210-3_195).
- 2187 [Rab83] M. O. Rabin. *Randomized Byzantine Generals*. In Proc. 24th Annual  
2188 Symposium on Foundations of Computer Science, SFCS '83, pages 403–409.  
2189 IEEE Computer Society, 1983. DOI:[10.1109/SFCS.1983.48](https://doi.org/10.1109/SFCS.1983.48).
- 2190 [Rad97] J. Radatz. *The IEEE Standard Dictionary of Electrical and Electronics Terms*.  
2191 IEEE Standards Office, 6th edition, 1997.
- 2192 [RSA78] R. L. Rivest, A. Shamir, and L. Adleman. *A method for obtaining digital*  
2193 *signatures and public-key cryptosystems*. Communications of the ACM,  
2194 21(2):120–126, 1978. DOI:[10.1145/359340.359342](https://doi.org/10.1145/359340.359342).
- 2195 [RSWO17] E. Ronen., A. Shamir, A.-O. Weingarten, and C. O'Flynn. *IoT Goes Nuclear:*

- 2196            *Creating a ZigBee Chain Reaction*. IEEE Symposium on Security and Privacy,  
2197            pages 195–212, 2017. DOI:[10.1109/SP.2017.14](https://doi.org/10.1109/SP.2017.14).
- 2198            [SA09] N. R. Sunitha and B. B. Amberker. *Forward-Secure Multi-signatures*.  
2199            In M. Parashar and S. K. Aggarwal (eds.), *Distributed Computing and*  
2200            *Internet Technology*, pages 89–99. Springer Berlin Heidelberg, 2009.  
2201            DOI:[10.1007/978-3-540-89737-8\\_9](https://doi.org/10.1007/978-3-540-89737-8_9).
- 2202            [Sau34] R. Saunders. *Poor Richard’s Almanack — 1735*. Benjamin Franklin, 1734.
- 2203            [Sch90] C. P. Schnorr. *Efficient Identification and Signatures for Smart Cards*.  
2204            In G. Brassard (ed.), *Advances in Cryptology — CRYPTO’89 Proceed-*  
2205            *ings*, vol. 435 of LNCS, pages 239–252. Springer New York, 1990.  
2206            DOI:[10.1007/0-387-34805-0\\_22](https://doi.org/10.1007/0-387-34805-0_22).
- 2207            [Sch99] B. Schoenmakers. *A Simple Publicly Verifiable Secret Sharing Scheme*  
2208            *and Its Application to Electronic Voting*. In M. Wiener (ed.), *Advances in*  
2209            *Cryptology — CRYPTO ’99*, vol. 1666 of LNCS, pages 148–164. Springer  
2210            Berlin Heidelberg, 1999. DOI:[10.1007/3-540-48405-1\\_10](https://doi.org/10.1007/3-540-48405-1_10).
- 2211            [Sha97] W. Shakespeare. *An excellent conceited Tragedie of Romeo and Juliet*. Printed  
2212            by John Danter, London, 1597.
- 2213            [Sha79] A. Shamir. *How to Share a Secret*. *Communications of the ACM*,  
2214            22(11):612–613, Nov 1979. DOI:[10.1145/359168.359176](https://doi.org/10.1145/359168.359176).
- 2215            [Sho00] V. Shoup. *Practical Threshold Signatures*. In B. Preneel (ed.), *Advances*  
2216            *in Cryptology — EUROCRYPT 2000*, vol. 1807 of LNCS, pages 207–220.  
2217            Springer Berlin Heidelberg, 2000. DOI:[10.1007/3-540-45539-6\\_15](https://doi.org/10.1007/3-540-45539-6_15).
- 2218            [SNV07] P. Sousa, N. F. Neves, and P. Verissimo. *Hidden Problems of Asynchronous*  
2219            *Proactive Recovery*. Proc. 3rd Workshop on on Hot Topics in System  
2220            Dependability, 2007.
- 2221            [Sta96] M. Stadler. *Publicly Verifiable Secret Sharing*. In U. Maurer (ed.), *Advances*  
2222            *in Cryptology — EUROCRYPT ’96*, vol. 1070 of LNCS, pages 190–199.  
2223            Springer Berlin Heidelberg, 1996. DOI:[10.1007/3-540-68339-9\\_17](https://doi.org/10.1007/3-540-68339-9_17).
- 2224            [SZ05] F. B. Schneider and L. Zhou. *Implementing Trustworthy Services Using*  
2225            *Replicated State Machines*. *IEEE Security and Privacy*, 3(5):34–43,  
2226            September 2005. DOI:[10.1109/MSP.2005.125](https://doi.org/10.1109/MSP.2005.125).
- 2227            [tC96] U. S. 104th Congress. *Information Technology Management*  
2228            *Reform Act. Public Law 104–106, Section 5131*, 1996. [https:](https://www.dol.gov/ocfo/media/regs/ITMRA.pdf)  
2229            [//www.dol.gov/ocfo/media/regs/ITMRA.pdf](https://www.dol.gov/ocfo/media/regs/ITMRA.pdf).
- 2230            [TJ11] H. C. A. Tilborg and S. Jajodia. *Encyclopedia of Cryptography and*  
2231            *Security*. Springer Publishing Company, Incorporated, 2nd edition, 2011.  
2232            DOI:[10.1007/978-1-4419-5906-5](https://doi.org/10.1007/978-1-4419-5906-5).

- 2233 [Vas15] A. Vassilev. *Cryptographic Validation Challenges With Brittle Algo-*  
2234 *rithms*. [https://csrc.nist.gov/groups/ST/lwc-workshop2015/presentations/](https://csrc.nist.gov/groups/ST/lwc-workshop2015/presentations/session5-vassilev.pdf)  
2235 [session5-vassilev.pdf](https://csrc.nist.gov/groups/ST/lwc-workshop2015/presentations/session5-vassilev.pdf), July 2015.
- 2236 [VCB<sup>+</sup>13] G. S. Veronese, M. Correia, A. N. Bessani, L. C. Lung, and P. Veríssimo.  
2237 *Efficient Byzantine Fault-Tolerance*. IEEE Transactions on Computers,  
2238 62(1):16–30, 01 2013. DOI:[10.1109/TC.2011.221](https://doi.org/10.1109/TC.2011.221).
- 2239 [vdGP88] J. van de Graaf and R. Peralta. *A Simple and Secure Way to Show the Validity*  
2240 *of Your Public Key*. In C. Pomerance (ed.), *Advances in Cryptology —*  
2241 *CRYPTO '87*, vol. 293 of LNCS, pages 128–134. Springer Berlin Heidelberg,  
2242 1988. DOI:[10.1007/3-540-48184-2\\_9](https://doi.org/10.1007/3-540-48184-2_9).
- 2243 [Ver18] Verizon. *2018 Data Breach Investigations Report*. [https://](https://www.verizonenterprise.com/verizon-insights-lab/dbir/)  
2244 [www.verizonenterprise.com/verizon-insights-lab/dbir/](https://www.verizonenterprise.com/verizon-insights-lab/dbir/), 2018.
- 2245 [VMB18] A. Vassilev, N. Mouha, and L. Brandão. *Psst, Can You Keep a Secret?* IEEE  
2246 Computer, 51(1):94–97, January 2018. DOI:[10.1109/MC.2018.1151029](https://doi.org/10.1109/MC.2018.1151029).
- 2247 [Yao82] A. C. Yao. *Protocols for secure computations*. In 23rd Annual Symposium  
2248 on Foundations of Computer Science, SFCS '82, pages 160–164, 11 1982.  
2249 DOI:[10.1109/SFCS.1982.88](https://doi.org/10.1109/SFCS.1982.88).
- 2250 [Yao86] A. C. Yao. *How to Generate and Exchange Secrets*. In Proc. 27th Annual  
2251 Symposium on Foundations of Computer Science, SFCS '86, pages 162–167.  
2252 IEEE Computer Society, 1986. DOI:[10.1109/SFCS.1986.25](https://doi.org/10.1109/SFCS.1986.25).

# List of feedback comments

The following pages show a table that includes public comments received about the draft published on July 26, 2018. Each source is indexed with an upper-case letter (A, B, ..., M), ordered chronologically by received date. There are 13 distinct sets of comments, plus one set of editorial revisions. Each set of comments was, for the purpose of this revision, separated in individualized items, e.g., A1, A2, ..., B1, ...

## Index

• <a href="#">A</a> — Svetla Nikova (KU Leuven) .....	60
• <a href="#">B</a> — Gokhan Kocak (Asena Inc.) .....	62
• <a href="#">C</a> — Oliver Stengele (Karlsruhe Institute of Technology) .....	63
• <a href="#">D</a> — Aivo Kalu (Cybernetica AS) .....	64
• <a href="#">E</a> — Christian Cachin, Hugo Krawczyk, Tal Rabin, Jason Resch, Chrysa Stathakopoulou (IBM, IBM Research) .....	66
• <a href="#">F</a> — Tanja Lange (Technische Universiteit Eindhoven) .....	82
• <a href="#">G</a> — Yehuda Lindell (Bar-Ilan University and Unbound Tech) .....	84
• <a href="#">H</a> — Samuel Ranellucci (Unbound Tech) .....	87
• <a href="#">I</a> — Dan Bogdanov (Cybernetica AS) .....	89
• <a href="#">J</a> — Rosario Gennaro (The City College of New York) .....	97
• <a href="#">K</a> — Thalia May Laing (HP Labs) .....	101
• <a href="#">L</a> — Karim Eldefrawy (SRI International) .....	104
• <a href="#">M</a> — John Wallrabenstein (Analog) .....	106
• <a href="#">N</a> — Authors (NIST) – editorial revision .....	108

#	Ref	Old location	A: Comments by Svetla Nikova (KULeuven)	Related	Reply Notes	Rev
1	A1	line 110, last words	“even if one” -> why do you write “one” for Example (ii) while Example (iii) has “some”?		<p>– <b>NOTE:</b> They are different examples, to convey a diversity of cases.</p> <p>– <b>CHANGED:</b> In the last example, changed “some” to “some (but not all)”.</p>	R5
2	A2	line 247, lines 983–984	<p>“f-out-of-n” threshold scheme</p> <p>I believe that this report is the first to introduce the parameter <math>f</math> for the number of nodes that can be tolerated to fail. The literature uses <math>k</math> and <math>n</math>, where the equality <math>k + f = n</math> holds for complete access structures, while incomplete (e.g. ramp not threshold) also exist where <math>k+f &lt; n</math> holds. This notation is explained only on line 983. Line 984 states that the introduction of <math>f</math> is useful, but the document does not make clear how. At least, the notation should be defined before it is used. I recommend to be careful with the terminology “f-out-of-n threshold scheme”, because readers might confuse it with the more commonly used term “k-out-of-n threshold scheme”.</p>	A2, E45, E60, F2, K11, K12	<p>– <b>NOTE:</b> The compromise-threshold parameter (<math>f</math> or some other letter) has been used in the literature, e.g., see Refs. [Lam06] and [GRJK00]. The subsequent paragraph (old lines 986–989) conveyed how both perspectives can be “useful”.</p> <p>– <b>CHANGED:</b> We have now added several clarifications about the co-existing dual (<math>f</math> vs. <math>k</math>) notation, e.g.: in section 1, new paragraph to mention <math>k</math>; old Sec 2.1 (“secret sharing”, now Sec. 2.2) now differentiates <math>f</math> and emphasizes <math>k</math>; old subsection 2.5 (“terminology”, now Sec. 2.1) now explicitly mentions the dual perspective; in Sec. 5.1.1, a slight text adjustment to bring “useful” closer to its explanation.</p>	R3, R13, R17, R18, R49, R74, R72
3	A3	Lines 714, 716, 719	you use $f_A$ , $f_I$ , $f_C$ but define them only on line 1012	A3, E61	<p>– <b>NOTE:</b> The symbols had been introduced in old lines 709–710 in Sec. 4.1, explaining that they correspond to different compromise thresholds for different properties.</p> <p>– <b>CHANGED:</b> Edited the mentioned paragraphs to clarify the index notation for different <math>f</math> thresholds. Move up (to Sec. 5.1.1) the note that clarifies the possible omission of indices.</p>	R51, R52, R53, R73
4	A4	Line 399-407	<p>is it useful to introduce the <math>(P, \omega)</math> notation in this report?</p> <p>Is it useful to mention an issue that is not a concern?</p>		<p>– <b>NOTE:</b> The angular (<math>\omega</math>) notation is used to convey complementary intuition for who may appreciate the geometric aspect of line rotation over a point (<math>P</math>).</p> <p>– <b>CHANGED:</b> Removed “concern”; simplified description.</p>	R29

#	Ref	Old location	A: Comments by Svetla Nikova (KULeuven)	Related	Reply Notes	Rev
5	A5	Line 622	[Ber05]: this paper discusses the effects of key-dependent cache hits vs. cache misses. I think this can also be called a memory access, so the statement that it is not sufficient to have source code without key dependent memory accesses is a bit dubious. Note also that the "remote attack" described in [Ber05] requires a malicious program to run locally and communicate with a remote host. The attack as described requires perfect knowledge of all software and libraries running on the system, in the correct versions. I think that the following is a better reference: Eran Tromer, Dag Arne Osvik, Adi Shamir: Efficient Cache Attacks on AES, and Countermeasures. J. Cryptology 23(1): 37-71 (2010)	<a href="#">A5</a> , <a href="#">E57</a>	<p>– <b>NOTE:</b> The old subsequent paragraph already explains/cites two use-cases where constant-time code may be insufficient.</p> <p>– <b>CHANGED:</b> For clarification: adjusted the position of the mentioned citation in the sentence; merged this to the subsequent paragraph, which already contained several references that justify the statement.</p>	<a href="#">R45</a>
6	A6	Line 1207	suggestion for an additional representative question: If a threshold scheme is proposed for a single-device setting, what assumptions does it make about the separation/independence of the "parties" that in practice sit next to one another in the device?	<a href="#">A6</a> , <a href="#">E19</a>	<p>– <b>NOTE:</b> The separation between parties is also a relevant matter in the single-device setting.</p> <p>– <b>CHANGED:</b> Added a related question.</p>	<a href="#">R93</a>
7	A7		The report mentions repeatedly that k-out-of-n threshold schemes, in particular when k is high, might lead to a lower security level than a 1-out-of-1 scheme. This is correct and needs to be mentioned, but I think it deserves less emphasis. In my opinion, the issue mentioned above is much more critical to assess the practical security of a threshold scheme.	<a href="#">A7</a> , <a href="#">G4</a> , <a href="#">E14</a> , <a href="#">E16</a> , <a href="#">J11</a>	<p>– <b>NOTE:</b> We find essential to highlight that there is a multitude of security properties that can be differently affected by a threshold augmentation. For example, if <math>k = n</math> then availability may be degraded. We agree that there are other critical issues.</p> <p>– <b>CHANGED:</b> No direct change, but see the reply to <a href="#">E14</a>.</p>	—

#	Ref	Old location	B: Comments by Gokhan Kocak (Asena Inc.)	Related	Reply Notes	Rev
8	B1		<p>I would like to share my observations about secret sharing schemes:</p> <ul style="list-style-type: none"> <li>- It's not publicly known</li> <li>- Even the IT Industry has limited knowledge about the secret sharing schemes</li> <li>- Storing pieces of data at different places is not an IT tradition, IT should accept this change and adapt to it.</li> </ul>	B1, G2, I2, M1	<p>– <b>NOTE:</b> This NISTIR intends to promote discussion, including with industry, about threshold schemes for cryptographic primitives, including those based on secret sharing schemes.</p> <p>– <b>CHANGED:</b> No change.</p>	—
9	B2		<p>We have implemented a threshold secret sharing using the following method:</p> <ul style="list-style-type: none"> <li>- Create a random key of sufficient length</li> <li>- Create a random nonce to be used in AES encryption</li> <li>- Encrypt the input using the random key</li> <li>- Create 3 pieces of the encrypted input using 2 out of 3 threshold scheme</li> <li>- Store the AES key, nonce, hash of the original input, hash of the encrypted output in a data structure of 512 bytes. Also add some padding with random data.</li> <li>- The hashes are used to make sure that the original input is created after combining pieces</li> <li>- Create 3 pieces of this data structure using 2 out of 3 threshold scheme</li> <li>- Try all possible combinations to re-create the original input in memory (i.e. one key is lost, one piece of data is lost, key 1 and key 3 are available, key 1 and key 2 are available etc)</li> <li>- Store all the pieces at different directories</li> <li>- Have the user distribute these pieces to different locations</li> </ul> <p>Algorithms we used:</p> <ul style="list-style-type: none"> <li>- Shamir's Secret Sharing algorithm</li> <li>- AES256 encryption</li> <li>- SHA256</li> </ul>		<p>– <b>NOTE:</b> The process of standardization should lead to clear descriptions of threshold schemes.</p> <p>– <b>CHANGED:</b> No change.</p>	—
10	B3		<p>We encrypted the original input because if someone finds 2 pieces of it, then he/she can re-create the encrypted data but not the original data. Because he/she needs the encryption key and nonce to decrypt it. We stored the hashes of input data and encrypted data in our key file together with AES key and nonce. After re-creating the original data, we used these hashes to check if we really created the original data. It's very important to keep the pieces of data and pieces of keys at different locations.</p>		<p>– <b>NOTE:</b> New standards of threshold schemes for cryptographic primitives should include techniques that enhance security.</p> <p>– <b>CHANGED:</b> No change.</p>	—

#	Ref	Old location	C: Comments by Oliver Stengele (KIT)	Related	Reply Notes	Rev
11	C1		<p>I am a scientific staff member within the research group „Decentralized Systems and Network Services“ at the Karlsruhe Institute of Technology (KIT) and I would like to offer a comment on your Draft NISTIR 8214 titled „Threshold Schemes for Cryptographic Primitives“.</p> <p>My current research focuses on using Threshold Encryption schemes to manage the flow of information in processes between mutually distrusting parties [2] using a decentralized consensus system like Ethereum [1]. Other researchers are using Threshold Encryption and Threshold Signature schemes, both with and without a trusted dealer, for other applications in the context of distributed systems [3,4].</p> <p>By their base functionality, Threshold Schemes appear destined to become a vital component of various distributed and decentralised processes. I would recommend that you keep this burgeoning field of possible applications in mind during the standardisation process.</p> <p>[1] V. Buterin, "A next-generation smart contract and decentralized application platform," white paper, 2014.</p> <p>[2] O. Stengele and H. Hartenstein, "Atomic Information Disclosure of Off-Chained Computations Using Threshold Encryption.," DPM/CBT@ESORICS, vol. 11025, no. 1, pp. 85-93, 2018.</p> <p>[3] E. Kokoris-Kogias, E. C. Alp, S. D. Siby, N. Gailly, P. Jovanovic, L. Gasser, and B. Ford, "Hidden in Plain Sight - Storing and Managing Secrets on a Public Ledger.," arXiv, 2018.</p> <p>[4] P. Schindler, A. Judmayer, N. Stifter, and E. Weippl, "Practical Continuous Distributed Randomness," eprint.iacr.org.</p>	C1, I2, I5, E30, H7 (higher-level apps)	<p>– <b>NOTE:</b> The report is focused on threshold schemes for cryptographic primitives (see old line 246). Threshold schemes can also fit a setting of decentralized trust. Higher-level apps can take advantage of threshold schemes.</p> <p>– <b>CHANGED:</b> Added two sentences in <a href="#">Sec. 1</a> (Introduction), referring to “decentralization of trust” and higher-level apps as complementary motivations.</p>	R11, R12

#	Ref	Old location	D: Comments by Aivo Kalu (Cybernetica)	Related	Reply Notes	Rev
12	D1		First, I would like to thank NIST for preparing the report on the threshold cryptography subject. This is a promising field with many possible applications and we are very interested to exchange views and have a discussion with you.	D1, E1, F1, G1, H2, H8, I1, J1, K1, L1, M2	<p>– <b>NOTE:</b> The endeavor has its complexities.</p> <p>– <b>CHANGED:</b> No change.</p>	—
13	D2		I would wish to highlight the possibility of the multi-prime RSA approach, which enables some rather interesting applications. As far as we know, multi-prime RSA is technically compliant with the RFC8017 definition and when the individual primes are generated according to the sizes and algorithms specified in the NIST FIPS 186-4, appendices B3, C.3, and F, it can be as secure as the two-prime RSA as well.		<p>– <b>NOTE:</b> The steps towards standardization of threshold schemes may analyze various approaches.</p> <p>– <b>CHANGED:</b> No change</p>	—
14	D3	Line 567	So, if we may propose a specific snippet to be added to the NISTIR text as well, the following place looks like a good candidate: Page 11, lines 567: "Different kind of approach, using multi-prime RSA, is suggested by [ <a href="https://www.researchgate.net/publication/297585554_On_the_Security_of_Distributed_Multiprime_RSA">https://www.researchgate.net/publication/297585554_On_the_Security_of_Distributed_Multiprime_RSA</a> ] and [ <a href="https://www.researchgate.net/publication/319071255_Server-Supported_RSA_Signatures_for_Mobile_Devices">https://www.researchgate.net/publication/319071255_Server-Supported_RSA_Signatures_for_Mobile_Devices</a> ], where the individual shares are generated as independent RSA key pairs and the public moduli of the shares are then multiplied together. This results in the dealer-less RSA key generation, without often expensive zero-knowledge proofs or SMPC protocols.		<p>– <b>NOTE:</b> In this paragraph we intended a close relation to the RSA example given in the beginning of section 3.1, which defined the modulus <math>N</math> as a product of only two primes.</p> <p>– <b>CHANGED:</b> Added a brief note about different RSA-based schemes allowing various tradeoffs, including related to properties of the modulus factorization.</p>	R41
15	D4		Also, the row "Bootstrap support" in the table 2 on the page 23 should be amended, to include the "multi-prime RSA" as well.	D4, E21	<p>– <b>NOTE:</b> Table 2 conveys several examples while, for the most part, keeping abstract from concrete primitives. For example, not even the two-prime RSA was included there.</p> <p>– <b>CHANGED:</b> In the first paragraph of Sec. 5, clarified that Table 2 does not intend to be exhaustive.</p>	R67

#	Ref	Old location	D: Comments by Aivo Kalu (Cybernetica)	Related	Reply Notes	Rev
16	D5		Another interesting issue with SplitKey signature scheme is the following consideration: At certain abstraction level, we are simply doing 2-out-of-2 or 3-out-of-3 threshold RSA signature scheme. However, the usual academic literature about the threshold schemes tends to assume that nodes, which hold the shares of the private key are independent and deploy the independent access control mechanism for consenting to the operation as well. Therefore, the "compromise" of the node might mean that the access control mechanism is compromised and attacker can therefore use the share to perform the crypto operation, even when he doesn't have the clear-text copy of the share.		<p>– <b>NOTE:</b> See reply to item <a href="#">D8</a>, about access control.</p> <p>– <b>CHANGED:</b> See reply to item <a href="#">D8</a>.</p>	—
17	D6		However, in my view, one of the important aspects of the SplitKey approach is that we have abstracted this 2-out-of-2 threshold access control of two nodes to the single end-user and we are providing him/her with a convenient way to use multiple authentication factors to control all nodes and this way, all shares. So, by entering the PIN (which will be used do decrypt the user's share) and by using the one-time-password (which is bound to the mobile device by updating it for each next operation) the access control to the local share and the shares, which are stored on the server-side, is achieved. Because this access control mechanism is interleaved tightly with keys and signatures, there's no way for the server to bypass this mechanism.		<p>– <b>NOTE:</b> See reply to item <a href="#">D8</a> about access control.</p> <p>– <b>CHANGED:</b> See reply to item <a href="#">D8</a>.</p>	—
18	D7		So, it may be difficult to explain the SplitKey signature scheme with all of the proposed characterizing features of Table 2 of NISTIR.	Diversity levels: <a href="#">D7</a> (AC), <a href="#">E16</a> (OS), <a href="#">E62</a> (CA), <a href="#">E71</a> (ref), <a href="#">E72</a> (vendors).	<p>– <b>NOTE:</b> Table 2 conveys examples of possible attributes of characterizing features, but does not intend to be exhaustive.</p> <p>– <b>CHANGED:</b> In Table 2, added a new row for "diversity levels", including "access control" as an example.</p>	<a href="#">R71</a>
19	D8	Lines 841-851	In the section 4.3 - System model you have already covered quite a lot of the details of the possible threshold systems. I would propose that the page 19, lines 841 - 851 also cover the possible ways to authenticate the users of the threshold system. It looks like that SplitKey approach, where the authentication is bound with some of the cryptography itself, may result a more secure system.		<p>– <b>NOTE:</b> Authentication between users and the threshold system is also a pertinent concern.</p> <p>– <b>CHANGED:</b> Several paragraphs below, under the "Trust between clients and threshold scheme" (within Sec. 4.3.3), added a short paragraph mentioning authentication, authorization and that an "access control" mechanism can also be thresholdized.</p>	<a href="#">R64</a>

#	Ref	Old location	E: Comments by Christian Cachin; Hugo Krawczyk; Tal Rabin; Jason Resch; Chrysoula Stathakopoulou (IBM)	Related	Reply Notes	Rev
20	E1		<p>We represent a group of researchers and practitioners within IBM who are working to create an open platform for threshold cryptography. We have many years of joint experience in developing threshold schemes and related technology.</p> <p>We want to extend our gratitude to you for starting this initiative to standardize threshold cryptography. We view threshold cryptography as an important tool in the fight against threats that face information systems. Standardizing the concepts, terminology, and metrics of threshold secure systems is a crucial first step in the path towards greater adoption of these techniques.</p>	<p><a href="#">D1</a>, <a href="#">E1</a>,  <a href="#">F1</a>, <a href="#">G1</a>,  <a href="#">H2</a>, <a href="#">H8</a>, <a href="#">I1</a>,  <a href="#">J1</a>, <a href="#">K1</a>, <a href="#">L1</a>,  <a href="#">M2</a></p>	<p>– <b>NOTE:</b> Thank you for the encouragement.</p> <p>– <b>CHANGED:</b> No change.</p>	—
21	E2		<p>In this shared goal of more widespread adoption of threshold cryptography, we hope to provide you with constructive feedback regarding the draft; a document we found to be impressively thorough and comprehensive. In this draft, you have put together a truly great and useful document. Thank you!</p>		<p>– <b>NOTE:</b> See <a href="#">E1</a>.</p> <p>– <b>CHANGED:</b> No change.</p>	—
22	E3		<p>Our high-level feedback is included below within this e-mail. It is broken down into one of six categories:</p> <ul style="list-style-type: none"> <li>Implementation</li> <li>Security Model</li> <li>Communications Channels</li> <li>Algorithms and Functions</li> <li>Post-Quantum Security</li> <li>Topics for Further Elaboration</li> </ul> <p>In addition to these high-level feedback, we have also attached to this e-mail a document listing a larger number minor items of feedback.</p>		<p>– <b>NOTE:</b> See comment items <a href="#">E4–E40</a>; to which will follow additional items <a href="#">E41–E72</a></p> <p>– <b>CHANGED:</b> No change.</p>	—

#	Ref	Old location	E: Comments by Christian Cachin; Hugo Krawczyk; Tal Rabin; Jason Resch; Chrysoula Stathakopoulou (IBM)	Related	Reply Notes	Rev
23	E4	Line 35	<p>Implementation</p> <p>In the introduction (line 35), it is mentioned that the Information Technology Laboratory develops proof of concept implementations. In the area of proof-of-concept implementations, we wanted to make you aware of our open source project ( currently hosted at <a href="https://github.com/jasonkresch/pross/">https://github.com/jasonkresch/pross/</a> ) which is an effort to build a back-end platform for threshold-secure distributed key generation, proactive refresh, share reconstruction, and rekeying of shareholders. Our aim is to extend its functionality with client-side protocols to enable threshold signing, decryption, password protected secret sharing, pseudo-random functions (including oblivious PRF), changes to K-of-N parameters, and other enhancements. It relies on a Byzantine-Fault Tolerant atomic broadcast channel provided by the BFT-SMaRt library.</p>		<p>– <b>NOTE:</b> Proof of concept implementations are of interest for this project.</p> <p>– <b>CHANGED:</b> No change.</p>	—
24	E5		<p>We view implementations as an important and necessary component of any standardization effort. Therefore, we consider it important to define requirements of implementations for which standards will be built. Some important questions to consider may include:</p>		<p>– <b>NOTE:</b> Implementations are important.</p> <p>– <b>CHANGED:</b> No change. See items E6–E12.</p>	—
25	E6		<p>Should standards submitters be required to provide working implementations?</p>		<p>– <b>NOTE:</b> Relevant.</p> <p>– <b>CHANGED:</b> Representative question 8a.</p>	R103
26	E7		<p>Should standards submitters demonstrate interoperability between two different implementations?</p>		<p>– <b>NOTE:</b> Relevant.</p> <p>– <b>CHANGED:</b> Representative question 8b.</p>	R104
27	E8		<p>Must implementations be open source, or should open source be viewed as a strong advantage?</p>		<p>– <b>NOTE:</b> Representative question 7a about licensing and intellectual property is general enough to cover many licensing models, including open-source.</p> <p>– <b>CHANGED:</b> No change.</p>	—
28	E9	3a	<p>Should implementations be designed for interoperability between hardware or operating systems? (for improved component diversity)</p>		<p>– <b>NOTE:</b> Relevant.</p> <p>– <b>CHANGED:</b> Representative question 3a.</p>	R94
29	E10		<p>Should the standard define common APIs for client-side functions (or should standards focus on back-end server inter-communication) – or might they be modular supporting a mix of different front- and back-end protocols?</p>		<p>– <b>NOTE:</b> Relevant.</p> <p>– <b>CHANGED:</b> Incorporated in the text.</p>	R95

#	Ref	Old location	E: Comments by Christian Cachin; Hugo Krawczyk; Tal Rabin; Jason Resch; Chrysoula Stathakopoulou (IBM)	Related	Reply Notes	Rev
30	E11		What degree of automated protocol analysis should be targeted for proposed standards? (During the TLS 1.3 standardization, many tools were used to prove protocols and implementations, including <a href="https://tamarin-prover.github.io">https://tamarin-prover.github.io</a> , <a href="http://prosecco.gforge.inria.fr">http://prosecco.gforge.inria.fr</a> , and <a href="https://www.microsoft.com/en-us/research/project/project-everest-verified-secure-implementations-https-ecosystem/">https://www.microsoft.com/en-us/research/project/project-everest-verified-secure-implementations-https-ecosystem/</a> )	E11, I8	<p>– <b>NOTE:</b> Relevant.</p> <p>– <b>CHANGED:</b> Incorporated in the text.</p>	R91, R96
31	E12		Should implementations demonstrate at least one high-level application (e.g. signing, decryption, PRF, etc.)		<p>– <b>NOTE:</b> Relevant.</p> <p>– <b>CHANGED:</b> Incorporated in the text.</p>	E12
32	E13		Security Model There is much discussion throughout the document on the topic of the security model of the threshold system and how it is related to the diversity of common components. On this topic we had a few high-level comments.		<p>– <b>NOTE:</b> Security model and diversity are important topics.</p> <p>– <b>CHANGED:</b> See items E14–E18.</p>	—
33	E14	Lines 117–119	In lines 117-119, the topic of whether in some cases using a distributed threshold system might increase an attack surface or otherwise might make some attacks easier than against a conventional (non-threshold) primitive. A chief example might be comparing a threshold system to a single location of a Hardware Security Module (HSM). HSMs are designed to make compromise as difficult as possible, and due to the limited functionality of most HSMs, one might have to chose between using a threshold system and using an HSM.	A7, G4, E14, E16	<p>– <b>NOTE:</b> Depending on the attack/compromise model, the security degradation can even happen if each of the components in the threshold scheme (e.g., several HSMs) is as resilient as the single component in the conventional scheme (e.g., a single HSM). The example suggested in the comment, comparing one HSM vs. a threshold system composed of several weaker components, and where the latter may be more secure, is a different (and also relevant) consideration.</p> <p>– <b>CHANGED:</b> Added text to the end of the <a href="#">paragraph</a>, to enable considering the two cases. See also the reply to E16.</p>	R7
34	E15		One thing that may be useful to highlight is that a number of off-the-shelf HSMs, which support only standard cryptographic functions, such as RSA signing, or ECDH key derivation, can in many cases be repurposed to perform threshold cryptography natively. This means for certain choices of threshold cryptography, one can get the advantages of threshold cryptography together with the high security of keeping shares only within an HSM.	E15, G4	<p>– <b>NOTE:</b> The 3-out-of-3 threshold RSA-signature example shows a case where each component does the same cryptographic operation as in a conventional operation. An actual threshold scheme may have to further include coordination between parties.</p> <p>– <b>CHANGED:</b> No change.</p>	—

#	Ref	Old location	E: Comments by Christian Cachin; Hugo Krawczyk; Tal Rabin; Jason Resch; Chrysoula Stathakopoulou (IBM)	Related	Reply Notes	Rev
35	E16		Using a system of K-out-of-N HSMs, from different HSM vendors, can also simplify the analysis of a security model. For example, by making some classes of operating system vulnerabilities irrelevant to the exposure of a share that is held on the HSM. One could construct a security model, borrowing from the ideas of reliability analysis where a "mean time to exposure" of a thresholdized secret could be derived from an estimated "mean time to exploit" and "mean time to patch" of the constituent components, assuming each of the N components (here an HSM) has an independent probability of an exploit being discovered and a fixed average time for a patch being issued and applied. In the case where leakage is silent, (undetected), this greatly amplifies the advantages of proactive refreshing.	Threshold security: <a href="#">A7</a> , <a href="#">G4</a> , <a href="#">E16</a> . Diversity levels: Diversity levels: <a href="#">D7</a> (AC), <a href="#">E16</a> (OS), <a href="#">E62</a> (CA), <a href="#">E71</a> (ref), <a href="#">E72</a> (vendors).	<p>– <b>NOTE:</b> Models can be used to compare the role of reactive and proactive refreshing.</p> <p>– <b>CHANGED:</b> In Sec. 4.1, added sentence mentioning that an analysis that models nodes of similar type (HSM, or other) having diversity at certain levels (e.g., OS, vendor), can help distinguishing the impact of diverse models of rejuvenation (<a href="#">R54</a>). In Sec. 5, added new row "diversity level" in Table 2, including examples "vendors" and "operating systems", among others (<a href="#">R71</a>).</p>	<a href="#">R54</a> , <a href="#">R71</a>
36	E17	Lines 1125–1135; Table 2	Secure Enclaves (e.g. SGX) are mentioned in the paragraph on lines 1125 - 1135. Perhaps these "secure enclaves" should be mentioned in Table 2 on page 23 as an example of software vs. hardware. Due to the limited programmability of most HSMs, secure enclaves (or any other trusted computing environment) provide a much needed layer within which to implement advanced protocols and functionality. While HSMs can hold and operate upon shares using PKCS11 functions, there is no built-in functionality for distributed key generation, proactive refresh, or share reconstruction. In non-programmable HSMs, this functionality must exist outside the confines of the HSM.	<a href="#">E15</a> , <a href="#">I19</a>	<p>– <b>NOTE:</b> Intermediate layers and hybrid instantiations enable potential solutions.</p> <p>– <b>CHANGED:</b> Added example "trusted computing environments" to the mentioned row of Table 2.</p>	<a href="#">R69</a>
37	E18		A final comment about the security model, is that there was limited discussion regarding the need for a trusted setup in these solutions. For example, in Pedersen's distributed key generation, there is a parameter "h" which no one should know the discrete logarithm of. Setting up such "nothing up my sleeve" numbers is a critical point of discussion in some threshold protocols.		<p>– <b>NOTE:</b> Section 3.1 already highlights differences between having a dealer and not having a dealer. Representative question <a href="#">4c</a> asks about trusted setup.</p> <p>– <b>CHANGED:</b> No direct change, but see new representative question <a href="#">4c</a> relating trusted setup and proofs of security.</p>	<a href="#">R101</a>

#	Ref	Old location	E: Comments by Christian Cachin; Hugo Krawczyk; Tal Rabin; Jason Resch; Chrysoula Stathakopoulou (IBM)	Related	Reply Notes	Rev
38	E19	Sec. 1	<p>Communications Channels</p> <p>Among other things, we were very pleased to see the level of attention that this document dedicates to considerations of the communications channel and how that can impact the security properties of the resulting threshold system. We consider it crucial enough to merit some discussion in the introduction (pages 1-3) as the attack models and realistic schemes differ enormously whether communication and the N components are (a) modules on the same chip; (b) components within the same server/machine; (c) servers in the same data center; (d) servers linked only by a wide-area network.</p>	A6, E19, E20, E21	<p>– <b>NOTE:</b> Attack models and security properties can vary substantially with the type of implementation platform and communication between nodes</p> <p>– <b>CHANGED:</b> In Sec. 1 (Introduction), mentioned this example after enumerating characterizing features.</p>	R15
39	E20	Lines 874–885	<p>This could be discussed more in section 4.3. The paragraph on lines 874-885 contains such a discussion that touches on these considerations, but it seems so important that it should be promoted to its own paragraph or subsection.</p>	E19, E20, E21, A6	<p>– <b>NOTE:</b> Sec. 4.3 motivates the need to describe system models; the later Sec. 4.3 discusses further aspects of its features.</p> <p>– <b>CHANGED:</b> For easier referencing the old paragraphs titles in subsection Sec. 4.3 were promoted to subsections with their own numbering.</p>	N5
40	E21	Table 2	<p>Some of the considerations also appear in Table 2 on page 23, but it is not exhaustive; the multiple parties vs. single device target executing platform lists only three possible deployments.</p>	D4, E21	<p>– <b>NOTE:</b> Table 2 does not intend to be exhaustive. Sec. 5.3.2 mentions a few other examples.</p> <p>– <b>CHANGED:</b> In the first paragraph of Sec. 5, clarified that Table 2 does not intend to be exhaustive.</p>	R67
41	E22	Sec. 5	<p>Overall, however, we find this section 5 "Characterizing features" to be highly useful and important.</p>		<p>– <b>CHANGED:</b> No change.</p>	—
42	E23	Lines 880–885	<p>We have very strong agreement with characterizing parameters listed, as starting on line 880. We think it is important to stress that many protocols described in the literature make assumptions (e.g. synchronized clocks or synchronous broadcast channels) that can be impractical or unrealistic in some deployments, such as over the Internet (which is more adequately modeled as an asynchronous network), and failing to consider such subtleties can result in complete breaks of the protocol's security.</p>		<p>– <b>NOTE:</b> Old lines 969–972 already mentioned that “if a system has all desired properties under a well-defined model, it may be unsuitable for real deployment if it fails catastrophically under reasonable variations of the environment” (Sec. 4.3.4).</p> <p>– <b>CHANGED:</b> No change.</p>	—

#	Ref	Old location	E: Comments by Christian Cachin; Hugo Krawczyk; Tal Rabin; Jason Resch; Chrysoula Stathakopoulou (IBM)	Related	Reply Notes	Rev
43	E24	Lines 1152...	On page 28, and the paragraph starting on 1152: another section should be inserted here, discussing the communication among the nodes (for example, using communication means (a)–(d) mentioned above).	E19	<p>– <b>NOTE:</b> The aspect of inter-node communication was already highlighted in old lines 1069–1077 (new Sec. 5.2.3). Considerations between “communication interfaces” (Sec. 5.2) and “computing platforms” (Sec. 5.3) are relevant.</p> <p>– <b>CHANGED:</b> No change.</p>	—
44	E25	Lines 158, 1423	Algorithms and Functions In several places, both lines 158 and 1423, the document expresses interest in enabling threshold versions of NIST-approved cryptographic primitives. We think it is important to consider also motivations flowing from the other direction. That is, when there are standards for threshold cryptography, NIST ought to consider standardizing cryptographic primitives that are threshold friendly (for example Schnorr and BLS signatures, ECIES decryption).	E25, E27, G2	<p>– <b>NOTE:</b> We have commented on this in old line 1424: “the consideration of standardization of threshold schemes is in itself an opportunity to review suitability for new standards.”</p> <p>– <b>CHANGED:</b> No change.</p>	—
45	E26		DSA/ECDSA, in particular has a number of properties which make it difficult or inefficient to thresholdize. (See <a href="https://eprint.iacr.org/2016/013.pdf">https://eprint.iacr.org/2016/013.pdf</a> for a treatment of this topic). ECDSA is increasingly replacing RSA, but no fully satisfactory threshold solution exists for it.		<p>– <b>NOTE:</b> Efficiency is a relevant property.</p> <p>– <b>CHANGED:</b> No change.</p>	—

#	Ref	Old location	E: Comments by Christian Cachin; Hugo Krawczyk; Tal Rabin; Jason Resch; Chrysoula Stathakopoulou (IBM)	Related	Reply Notes	Rev
46	E27	Line 1398	<p>In terms of the request on line 1398, "to develop an objective criteria set to support a call for and a selection of schemes for standardization", we list below a set of schemes that are both simple and efficient to thresholdize:</p> <ul style="list-style-type: none"> <li>Distributed Secret Generation, Storage, Retrieval</li> <li>Long term storage of keys, or short small secrets</li> <li>Threshold PRF</li> <li>More secure/trusted/harder to bias PRF, KeyStream generation</li> <li>Threshold (Partially) Oblivious PRF</li> <li>Password protected secret sharing (<a href="https://eprint.iacr.org/2017/363">https://eprint.iacr.org/2017/363</a>), password databases (<a href="https://eprint.iacr.org/2015/644.pdf">https://eprint.iacr.org/2015/644.pdf</a>), key management systems (<a href="https://eprint.iacr.org/2018/733">https://eprint.iacr.org/2018/733</a>), password management (<a href="https://eprint.iacr.org/2018/695.pdf">https://eprint.iacr.org/2018/695.pdf</a>), PAKEs (<a href="https://eprint.iacr.org/2018/163.pdf">https://eprint.iacr.org/2018/163.pdf</a>)</li> <li>Threshold (RSA, Schnorr, BLS) Signatures</li> <li>Distributed Certificate Authorities</li> <li>Threshold Bilinear Pairing</li> <li>Identity based encryption, BLS signatures, Threshold Partially Oblivious PRFs</li> <li>Threshold (RSA, El Gamal, ECIES) Decryption</li> <li>Voting schemes, long term storage of sensitive documents to be later published (e.g. wills, audit records).</li> </ul>	E25, E27, G2	<p>– <b>NOTE:</b> We hope to hear from stakeholders about recent and other threshold schemes. The representative questions in Sec. 7.1 suggest several aspects to considered when discussing those schemes.</p> <p>– <b>CHANGED:</b> No change.</p>	—
47	E28		Notably, some of these schemes do not even have non-threshold (1-of-1) standardization.	E25	<p>– <b>NOTE:</b> See reply to item E25</p> <p>– <b>CHANGED:</b> No change.</p>	—
48	E29		There is also the matter of standardizing the maintenance operations (protocols to refresh and reconstruct shares over time).		<p>– <b>NOTE:</b> “Maintenance” is identified in a main domain of characterizing features (see “Setup and Maintenance” in Sec. 5.4), namely including considerations about “rejuvenation of nodes” Sec. 5.4.2.</p> <p>– <b>CHANGED:</b> No change.</p>	—
49	E30		We think it would be helpful to include a section for discussing various applications of different threshold operations; at a minimum it could serve to motivate further research and adoption.	C1, I2, I5, E30, H7 (higher-level apps)	<p>– <b>NOTE:</b> Applications can be motivating. We hope to hear more from stakeholders.</p> <p>– <b>CHANGED:</b> See reply to I5.</p>	R12

#	Ref	Old location	E: Comments by Christian Cachin; Hugo Krawczyk; Tal Rabin; Jason Resch; Chrysoula Stathakopoulou (IBM)	Related	Reply Notes	Rev
50	E31	Line 1427, Sec. 7	Post-Quantum Security On line 1427 there is a brief treatment of post-quantum cryptography. On this topic we believe NIST could be enormously helpful in motivating research into post-quantum threshold-secure, cryptography. There has been some research into threshold systems for lattice-based cryptography ( <a href="https://eprint.iacr.org/2009/391.pdf">https://eprint.iacr.org/2009/391.pdf</a> , <a href="https://www.tau.ac.il/~tromer/papers/tfhe-mpc.pdf">https://www.tau.ac.il/~tromer/papers/tfhe-mpc.pdf</a> ) as well as a recent "Universal Thresholdizer" <a href="https://eprint.iacr.org/2017/251">https://eprint.iacr.org/2017/251</a> . However, more research is required in this area for constructing threshold systems based on PQ primitives such as lattices, isogeny cryptography, etc.	E31, E32	<p>– <b>NOTE:</b> The consideration of PQ threshold schemes is natural, since both PQ schemes and threshold schemes are developing interests.</p> <p>– <b>CHANGED:</b> In Sec. 7.3, in the corresponding paragraph, we added a short note: "More research is needed".</p>	R111
51	E32		One point which would be prudent to consider now is whether PQ algorithms to be selected by NIST for standardization should include "efficient thresholdizability" as a desirable criterion.	E31, E32	<p>– <b>NOTE:</b> See reply to item E31.</p> <p>– <b>CHANGED:</b> No change.</p>	R111
52	E33		Topics for Further Elaboration This section covers various areas, and is meant to highlight areas we thought could benefit from more elaboration.	E34–E40	<p>– <b>NOTE:</b> See items E34–E40.</p> <p>– <b>CHANGED:</b> No change.</p>	—
53	E34		While much time is spent discussing RSA, there is no mention of elliptic curves (or Diffie-Hellman-type schemes) which use math and keys that are efficient and straightforward to apply in a threshold scheme. More modern algorithms and techniques, which are also threshold-friendly, such as bilinear pairings, PRFs/OPRFs, are similarly absent. It might be useful to include a table, denoting different key types and algorithms with known efficient threshold solutions.	E34, F4	<p>– <b>NOTE:</b> The report covers only a few examples. Benchmarking may be useful ahead.</p> <p>– <b>CHANGED:</b> No direct change, but see mention to benchmarking in the new last paragraph in Sec. 7.2.</p>	—
54	E35		Along these lines, there was no discussion of the relationship between the shares generated of a key, and the cryptosystem in question. For example, how should the prime Q be selected when using NIST P-256 vs. secp521r1. When different, this means shares used for EC group should not be employed if the prime Q differs between the two cryptosystems.	E35	<p>– <b>NOTE:</b> The characteristics of threshold schemes may depend on characteristics (e.g., cryptographic parameters) of the underlying cryptographic scheme.</p> <p>– <b>CHANGED:</b> No change.</p>	—

#	Ref	Old location	E: Comments by Christian Cachin; Hugo Krawczyk; Tal Rabin; Jason Resch; Chrysoula Stathakopoulou (IBM)	Related	Reply Notes	Rev
55	E36		It might also be prudent for the document to suggest that the threshold number of servers should restrict "key usages" for each of the shares they hold, e.g., to prevent shares of a signing key from being used in a decryption operation.		<p>– <b>NOTE:</b> Guidelines for key usage are relevant also for threshold schemes.</p> <p>– <b>CHANGED:</b> No change.</p>	—
56	E37		An important, but insufficiently discussed topic was a detailed consideration of crash and compromise recovery procedures. For example, after a compromise a share refresh is often not sufficient. A node may have used a private key to identify itself to other shareholders, and if this key is compromised it needs to be replaced. Whether this involves a CA, or a trusted administrative intervention, or manual updates to each of the N-1 other shareholders is an important distinction between various implementations.		<p>– <b>NOTE:</b> Aspects of maintenance (Sec. 5.4) are relevant for threshold schemes, including with respect to the type of rejuvenation (Sec. 5.4.2)</p> <p>– <b>CHANGED:</b> No change.</p>	—
57	E38		The related problem of recovery is also important and extends beyond share reconstruction: for example in the replicated-state-machine sense, it might require replaying all previously broadcast messages on the shared channel to re-synchronize the state of the recovered shareholder. Alternatively, it might require receiving a representation of the current state signed (or otherwise attested to) by at least F+1 shareholders.		<p>– <b>NOTE:</b> The report also considers rejuvenation of stateful nodes: stateless vs. stateful rejuvenation modes (old line 1239); rejuvenation of stateful nodes may require specifying a sub-protocol (old lines 1248–1250).</p> <p>– <b>CHANGED:</b> No change.</p>	—
58	E39	Line 1216	While there is mention of distributed key generation and "dealer free" setups, we felt insufficient time was dedicated to this topic. For example, the paragraph at line 1216 discusses dealer-based setup at length but perhaps should also cover distributed key generation protocols (For example: Rosario Gennaro, Stanislaw Jarecki, Hugo Krawczyk, Tal Rabin: Secure Distributed Key Generation for Discrete-Log Based Cryptosystems. J. Cryptology 20(1): 51-83 (2007))	E39, F4	<p>– <b>NOTE:</b> Distributed Key generation is a relevant topic. For example, Sec. 3.1 cites distributed key-generation protocols for RSA-based and discrete-log based schemes [Ped91, BF97].</p> <p>– <b>CHANGED:</b> In Sec. 3.1.4, added Ref. [GJKR99] that reviews some aspects of [Ped91].</p>	R43
59	E40		While the document does discuss loss of availability, there was little discussion of denial of service, and what special precautions or advantages exist in threshold cryptography systems.	E40, J12–J14	<p>– <b>NOTE:</b> Threshold schemes also provide tradeoffs for protection of availability.</p> <p>– <b>CHANGED:</b> In Sec. 4.1.3, added paragraph about attacks on availability.</p>	R57
60	E41		(additional feedback items are included in the attached document: "Additional-Comments-NIST-8214.rtf")		<p>– <b>NOTE:</b> See items E42–E72.</p> <p>– <b>CHANGED:</b> No change.</p>	—

#	Ref	Old location	E: Comments by Christian Cachin; Hugo Krawczyk; Tal Rabin; Jason Resch; Chrysoula Stathakopoulou (IBM)	Related	Reply Notes	Rev
61	E42	Lines 106-107	"This is possible to achieve for NIST-approved algorithms, such as RSA and DSA signatures, and AES enciphering and deciphering." > A citation to the work regarding thresholdized AES encryption and decryption would be instructive. For example: <a href="https://eprint.iacr.org/2018/727.pdf">https://eprint.iacr.org/2018/727.pdf</a>	<a href="#">E56, J2</a>	– <b>NOTE:</b> In Sec. 2.4 we have conveyed that any threshold scheme can be implemented using SMPC. (See also last paragraph of Sec. 1.) – <b>CHANGED:</b> No direct change, but see reply to items <a href="#">E56</a> and <a href="#">J2</a> , about SMPC enabling threshold schemes for cryptographic primitives.	—
62	E43	Lines 122-123	"the corruption of a single share (or of a computation dependent on it) may affect the integrity of the output" > The particular means of verification of partial results can vary quite substantially from scheme to scheme, and depends largely on the cryptographic algorithm and function being computed. Some discussion on these various mechanisms, or citations providing some examples would be useful. Sometimes individual components can be efficiently checked (e.g. as a RSA signature verify operation), other times it may be easier to check the consistency of the final result first, and only fall back to individual verification if that top-level check fails.	<a href="#">K2–K6</a>	– <b>NOTE:</b> The executive summary intends to be very high-level. – <b>CHANGED:</b> No change, but see reply to <a href="#">K6</a> .	—
63	E44	Lines 122-123	> In one example threshold schemes support a transparent verification via an "interactive proof", (See section 4 of <a href="https://eprint.iacr.org/2018/733.pdf">https://eprint.iacr.org/2018/733.pdf</a> ), in another example a client can use a "BLS signature" to prove each server (or the overall combined value) is consistent, at least when operating over a pairing-friendly curve.	<a href="#">E43</a>	– <b>NOTE:</b> Different threshold schemes may achieve verifiability properties via different techniques. – <b>CHANGED:</b> No change.	—
64	E45	Line 332,333	"scheme has a “3-out-of-3” property." > This is a redefinition from the previously given definition of an "f-out-of-n" system where f is the maximum number of tolerated faults. When only numerals are provided, there is no clarity between whether the text is describing a k-out-of-n system or an f-out-of-n system. It might be better to standardize on a single convention.	<a href="#">A2, E45, E60, F2, K11, K12</a>	– <b>NOTE:</b> We agree a clarification here ( <i>k</i> for an operational requirement of number of good participating parties; <i>f</i> for number of tolerated faults/compromises) can be helpful. We see value in being able to allude to two types of threshold, provided the context is clear. – <b>CHANGED:</b> Edited the paragraph to make explicit which kind of threshold has which value: $f = 2, k = 3$ . See also the reply to <a href="#">A2</a> .	<a href="#">R3, R13, R17, R18, R49, R74, R72</a>

#	Ref	Old location	E: Comments by Christian Cachin; Hugo Krawczyk; Tal Rabin; Jason Resch; Chrysoula Stathakopoulou (IBM)	Related	Reply Notes	Rev
65	E46	Line 354	"The coefficients $h_i$ are non-zero" > Must the coefficients be non-zero? Shouldn't they be uniform modulo $Q$ ( $0 < Q-1$ )? The critical property is that any $k$ of the hyperplane equations be linearly independent.	N6	<p>– <b>NOTE:</b> Linear independence follows from requiring that any subset of <math>k</math> hyperplanes (each of dimension <math>k - 1</math>) in the <math>k</math>-dimensional space intersect in a single point. Besides linear independence, it is also necessary to require that all planes are not orthogonal to the axis of projection (e.g., <math>x_1</math>) used to derive the secret from the point <math>P</math>.</p> <p>– <b>CHANGED:</b> Replaced the incorrect requirement of non-zero coefficients <math>h_i</math>, by a new requirement (see item N6).</p>	R20
66	E47	Line 357	"larger than the number $n$ of parties" This is the only discussion regarding limits on the number of parties, or for that matter any limitations or considerations on the limits of $n$ , $k$ , or $f$ . The document might benefit from some discussion on the optimal selection of these parameters, and necessary relations imposed between them. Also to consider: are there any or certain schemes that break down for very large values of $n$ ?	E47, E51	<p>– <b>NOTE:</b> In old line 357, the indicated inequality is a simple statement of a condition for the purpose of a secret sharing scheme. Sec. 5.1 makes additional considerations about parameters <math>k</math>, <math>f</math> and <math>n</math>. Optimal choices may vary with the threshold scheme, the attack model, and cost considerations.</p> <p>– <b>CHANGED:</b> No change.</p>	
67	E48	Line 385	"Shamir secret sharing is based on the observation that any set of $k$ distinct points" > While the math of Shamir is typically described in terms of polynomial interpolation, while Blakley's scheme is described in terms of hyperplane geometry, both schemes are effectively identical in terms of how shares are computed, represented, and recovered. The operative difference is that Shamir is just specific about how the " $h$ coefficients" are determined (as according to a Vandermonde matrix, thereby guaranteeing any subset of $k$ equations are linearly independent). Highlighting this similarity (that Shamir is a special case of Blakley) may be beneficial for readers unfamiliar with them. The important implication is that anything possible with the shares of one scheme is by definition possible to do using shares of the other scheme.		<p>– <b>NOTE:</b> Old lines 334-336 (currently the third paragraph of Sec. 2.2) already describe the intended capability (<math>k</math>-out-of-<math>n</math> secret-sharing) achieved by both schemes: "any <math>k</math> parties together can recover a secret shared across <math>n</math> parties, but <math>k - 1</math> parties together do not know anything about the secret."</p> <p>– <b>CHANGED:</b> No change, but added paragraph titles to separate better the description of the two schemes.</p>	R23
68	E49	Line 369	"less than $k$ " > Should be "fewer than $k$ "		<p>– <b>NOTE:</b> Agreed.</p> <p>– <b>CHANGED:</b> As suggested.</p>	R24

#	Ref	Old location	E: Comments by Christian Cachin; Hugo Krawczyk; Tal Rabin; Jason Resch; Chrysoula Stathakopoulou (IBM)	Related	Reply Notes	Rev
69	E50	Lines 371-372	"This means that the scheme can in practice be used to share very small secrets (e.g., only a few bits)," > "Only a few bits" seems unjustified. Computers have little difficulty operating over fields defined by primes of hundreds or thousands of bits, which is much larger than "only a few". Other language besides "few bits" and "very small secrets" might be better, for example "secrets on the order of a few hundred bytes or less". The real limitation therefore is imposed by the inefficiency of having to store N times as much data as the size of the secret, rather than the size of the finite field in which the math is applied.		<p>– <b>NOTE:</b> The intended emphasis is on conveying that security (in an information theoretical sense) does not depend on the size of the prime or of the secret, provided the relation of sizes holds.</p> <p>– <b>CHANGED:</b> To minimize confusion, we relegated the “smallness” aspect, so that the information theoretical aspect is more salient.</p>	R25
70	E51	Line 383	"Secret resharing" > In addition to resharing: there are the related concepts of adding shareholders, removing shareholders, replacing shareholders, changing N, changing K, reconstructing lost shares, and rekeying compromised shareholders.	E47, E51, E54	<p>– <b>NOTE:</b> Changes in the threshold set of nodes is a relevant matter and may require its own sub-protocols.</p> <p>– <b>CHANGED:</b> In Sec. 3.1.3, the new paragraph about resharing mentions the possibility of changing the threshold structure, including changing the number of parties.</p>	R44
71	E52	Line 388	> Beyond resharing, there exists the necessary steps to patch/secure/reboot/reset/rekey systems in when the need for rekeying is based on a detected compromise.		<p>– <b>NOTE:</b> Sec. 2.3 is building up on the concept of secret sharing; other rejuvenation aspects are considered in Sec. 5.4.2.</p> <p>– <b>CHANGED:</b> No change.</p>	—
72	E53	Line 391	> While correct for this example, it might be worth stating that generally, the protocol would re-randomize all coordinates except for the one that represents the secret. As written it sounds as if only one of the coordinates needs to be re-randomized, but really it is (K-1) of the coordinates (or coefficients in Shamir)	N7, E53	<p>– <b>NOTE:</b> The paragraph in question was scoped to <math>k = 2</math>.</p> <p>– <b>CHANGED:</b> Added short related sentence in Sec. 2.3.</p>	R31
73	E54	Line 416	> As described this is describing the operation in the case of a central dealer resharing the same secret. But there are substantial differences when resharing in a dealerless protocol. If the details of this are not covered, it may be worth mention that protocols exist to perform resharing in a dealerless configuration.	E54, F3	<p>– <b>NOTE:</b> Agreed.</p> <p>– <b>CHANGED:</b> See reply to item F3.</p>	R44

#	Ref	Old location	E: Comments by Christian Cachin; Hugo Krawczyk; Tal Rabin; Jason Resch; Chrysoula Stathakopoulou (IBM)	Related	Reply Notes	Rev
74	E55	Line 437,438	> It seems appropriate to add [BGW] to the citation of Yao and GMW. These works provide approaches with different performance tradeoffs and applicability. As an example, recent performance optimizations of Yao's garbled circuit technique lead to protocols that are relatively efficient in computation but heavy in computation. This presents an example where the LAN/WAN distinction may be crucial in choosing the best implementation. For the LAN setting, Yao may be best, but over WAN one would choose BGW (or GMW).	E55, I11	<p>– <b>NOTE:</b> SMPC can be achieved in a variety of settings.</p> <p>– <b>CHANGED:</b> Added early references to unconditionally secure SMPC.</p>	R32
75	E56	Line 448	> It might be more correct to say that all threshold schemes are special cases of SMPC.	E56, J2	<p>– <b>NOTE:</b> We consider SMPC under the typical framework of ideal functionalities and protocols that emulate an ideal functionality. Some systems may internally implement a threshold approach (e.g., secret-sharing, replication) while not being conceptualized in an SMPC manner.</p> <p>– <b>CHANGED:</b> Adjustments to section 2.4, clarifying the intended distinction.</p>	R33, R34, R35
76	E57	Line 464	> On the subject of side-channel attacks, some consideration for designing implementations and algorithms for threshold cryptography (exponentiation on shares, share generation, etc.) should be designed for constant-time operation. Otherwise information about the share could be leaked over time.	A5, E57	<p>– <b>NOTE:</b> In old lines 618–626 we already covered the relevance of constant-time operations.</p> <p>– <b>CHANGED:</b> No change.</p>	R45
77	E58		Periodic proactive resharing may be an effective countermeasure, especially if applied after so many operations involving a particular share.		<p>– <b>NOTE:</b> Proactive resharing is mentioned across the report.</p> <p>– <b>CHANGED:</b> No change.</p>	—
78	E59	Lines 676-685	> Thresholdizing a key before performing a computation may yield side-channel attack benefits even in a single computation performed on one processor. That is, by randomizing the secret information on shares upon which the computation is performed. Similar to how some implementations mask timing information in RSA by incorporating a random factor which gets factored out.	E59, M7	<p>– <b>NOTE:</b> Several threshold approaches are of interest, including for the single-device setting. Sec. 2.5 mentions some approaches applicable to address side-channel attacks in the single-device setting.</p> <p>– <b>CHANGED:</b> No change.</p>	—

#	Ref	Old location	E: Comments by Christian Cachin; Hugo Krawczyk; Tal Rabin; Jason Resch; Chrysoula Stathakopoulou (IBM)	Related	Reply Notes	Rev
79	E60	Lines 708-710	> In general $f$ and $k$ may make poor choices for describing threshold schemes, if only because $f$ and $k$ are so commonly used to represent other things. "t" for threshold might be better. The use of subscripts for indicating what kind of threshold is involved is also a good strategy. Perhaps "f" could instead be $t$ , such that " $t_s$ " (the degree of the polynomial) is the threshold of maintaining secrecy, and $t_s+1$ is the threshold for exposing the secret ( $t_e$ ), as an example.	<a href="#">A2</a> , <a href="#">E45</a> , <a href="#">E60</a> , <a href="#">F2</a> , <a href="#">K11</a> , <a href="#">K12</a>	<p>– <b>NOTE:</b> We had exemplified the use of subscripts when differentiating thresholds across different security properties. When focusing on a property, e.g., confidentiality (C), identified in subscript, we still use a different symbol (<math>f</math> vs. <math>k</math>) to distinguish the type of threshold (compromise threshold <math>f</math> vs. operational requirement of needed number of participating parties <math>k</math>).</p> <p>– <b>CHANGED:</b> We made several clarifications in the document. See reply to <a href="#">A2</a>.</p>	<a href="#">R3</a> , <a href="#">R13</a> , <a href="#">R17</a> , <a href="#">R18</a> , <a href="#">R49</a> , <a href="#">R74</a> , <a href="#">R72</a>
80	E61	Line 710	> There could perhaps be a little more clarification regarding $F_i 0$ vs. infinity. Perhaps an example would help.	<a href="#">A3</a> , <a href="#">E60</a> , <a href="#">E61</a>	<p>– <b>NOTE:</b> Agreed.</p> <p>– <b>CHANGED:</b> Changed <math>f_I = \infty</math> to <math>f_I = n</math> and improved explanation.</p>	<a href="#">R52</a>
81	E62	Line 897	> Another possibility when using PKI is for full threshold security (no single points of compromise), one might consider having $N$ different CAs, each one responsible and recognized as the only CA for issuing certificates of a given node's shareholder index.	Diversity levels: <a href="#">D7</a> (AC), <a href="#">E16</a> (OS), <a href="#">E62</a> (CA), <a href="#">E71</a> (ref), <a href="#">E72</a> (vendors).	<p>– <b>NOTE:</b> It is important to consider diversity levels.</p> <p>– <b>CHANGED:</b> Added example "CA" in the new row "diversity levels" of Table 2.</p>	<a href="#">R71</a>
82	E63	Line 959	> This is potentially an overloading of $k$ . It is conceivable that a system could have a different recovery threshold for the secret sharing scheme than the $(n - f)$ good nodes that the byzantine fault tolerant communication channel requires.		<p>– <b>NOTE:</b> The meaning of "good" (resp. "bad") components depends on what it applies to. A threshold scheme for a cryptographic primitive may be based on a secret-sharing scheme requiring <math>f + 1</math> good shares to reconstruct a secret, whereas the overall threshold scheme may require that <math>2f</math> nodes remain uncompromised, e.g., possibly due to inability to distinguish between delayed vs. faulty nodes. When need-be the thresholds can be distinguished with additional indices.</p> <p>– <b>CHANGED:</b> No change.</p>	—
83	E64	Line 960	> A non-crashed simple-majority is not sufficient yet (for consensus, because of FLP85 but the protocols need additional timing assumptions to ensure liveness).		<p>– <b>NOTE:</b> Old line 953 did mention (with citation <a href="#">[FLP85]</a>) that the problem is unsolvable in a deterministic and asynchronous setting. The statement in old line 960 intended to retain the scope of access to randomness and timing assumptions (synchronous communication) mentioned a few lines before (old lines 956–958).</p> <p>– <b>CHANGED:</b> In Sec. 4.3.4, clarified statement by re-mentioning the timing assumptions.</p>	<a href="#">R65</a>

#	Ref	Old location	E: Comments by Christian Cachin; Hugo Krawczyk; Tal Rabin; Jason Resch; Chrysoula Stathakopoulou (IBM)	Related	Reply Notes	Rev
84	E65	Page 23, Table 2	> Consider adding next to the TLS entry "self" (meaning the protocol itself handles the security of the communications, e.g., via direct encryption under public keys).	E65, I14, K10	<p>– <b>NOTE:</b> Channel security can be substantiated in various manners.</p> <p>– <b>CHANGED:</b> Added examples to the corresponding row in Table 2.</p>	R68
85	E66	Line 981	> There is a further notion of <code>_dual_</code> threshold schemes. They introduce a third parameter <code>L</code> (in the paper it is <code>k</code> ), which says from how many nodes a contribution is needed to reconstruct. This can be larger than <code>f</code> ; for example $L = 2 * f + 1$ . See: Christian Cachin, Klaus Kursawe, Victor Shoup: Random Oracles in Constantinople: Practical Asynchronous Byzantine Agreement Using Cryptography. J. Cryptology 18(3): 219-246 (2005)		<p>– <b>NOTE:</b> The possibility that <math>k</math> is larger than <math>f + 1</math> is allowed by our notation. This is common when <math>n \geq 3f + 1</math>.</p> <p>– <b>CHANGED:</b> No change.</p>	—
86	E67	Line 1034,1035	> Zero Knowledge Proofs (and interactive proofs) can be used not only for verifying correctness of operations and partial contributions to the output of a cryptographic function, but are also critically important to some distributed key generation and resharing protocols. Aside from such proofs, some functionalities are self-verifiable (e.g signatures). Another possibility is using pairings and BLS (when specific pairing friendly curves are used), with the advantage that there is no overhead for the server to produce a proof.		<p>– <b>NOTE:</b> ZKPs have wide applicability. The paragraph ended with an example where ZKPs could be useful for the matter discussed in this paragraph. This paragraph does not focus on concrete schemes.</p> <p>– <b>CHANGED:</b> No change.</p>	—
87	E68	Line 1038	> It would be appropriate to cite also the paper on Byzantine quorum systems that predates HM00: Dahlia Malkhi, Michael K. Reiter: Byzantine Quorum Systems. Distributed Computing 11(4): 203-213 (1998)		<p>– <b>NOTE:</b> Various works discuss notions mentioned in this report. We have two references focused on arbitrary access structures (one from 1989).</p> <p>– <b>CHANGED:</b> No change.</p>	—
88	E69	Line 1079	"Are clients aware of the threshold nature of the implementation?" > Perhaps this property can be referred to as "transparency" and highlighted as an important feature in many settings. The definition of transparency being that the user of the service cannot distinguish between a threshold implementation and a centralized service. (This is mentioned section 5.2 but the transparency advantage is not highlighted). There is also the question of whether ZK proofs of correctness can be verified transparently (e.g., combined from each of the threshold server's individual ZK or interactive proofs by a service aggregator or proxy).	E69, E70, I12	<p>– <b>NOTE:</b> The property suggestively called "transparency" also conveys "opacity", since it equates to hiding the threshold structure from the client. The property can be relevant; in some cases the opposite is also an advantage, e.g., in multi-signatures. Sec. 5.2 is about communication interfaces, but the suggested property is not only about communication interfaces. The question about ZK may be specific to concrete threshold schemes. See complementary observation in item I12.</p> <p>– <b>CHANGED:</b> No change.</p>	—

#	Ref	Old location	E: Comments by Christian Cachin; Hugo Krawczyk; Tal Rabin; Jason Resch; Chrysoula Stathakopoulou (IBM)	Related	Reply Notes	Rev
89	E70	Line 1261	> The notion of a "primary node" should not be exposed by a threshold scheme, it should always be hidden and dealt with inside. Otherwise, it does not satisfy the notion that "any f" nodes could fail. For example, protocols like PBFT [CL02] automatically elect a new leader and this must be totally transparent from outside.	E69, E70, I12	<p>– <b>NOTE:</b> It can still satisfy the notion of <math>f</math> for some property, e.g., key confidentiality if the key is shared across all nodes. While non-exposure of the primary may be advantageous in some settings, there are conceivable settings where it may be advantageous to have it exposed. For example, a setting that requires a full audit-trail of the threshold process could want to identify who was the leader during a threshold execution.</p> <p>– <b>CHANGED:</b> Slight adjustment in <a href="#">sentence</a> about communication via primary node.</p>	R77
90	E71	Line 1270	"Levels of diversity" > The critical issue of how to achieve diversity in a security context is discussed in depth Schneider and Zhou. This should be cited or material from there included. Fred B. Schneider, Lidong Zhou: Implementing Trustworthy Services Using Replicated State Machines. IEEE Security & Privacy 3(5): 34-43 (2005)	Diversity levels: D7 (AC), E16 (OS), E62 (CA), E71 (ref), E72 (vendors).	<p>– <b>NOTE:</b> It is important to consider diversity levels.</p> <p>– <b>CHANGED:</b> Added row “diversity levels” to Table 2; added suggested reference that surveys the use of diversity.</p>	R71, R88
91	E72	Line 1291	> Another consideration: a limited number of variants for HSM vendors.	Diversity levels: D7 (AC), E16 (OS), E62 (CA), E71 (ref), E72 (vendors).	<p>– <b>NOTE:</b> It is important to consider diversity levels.</p> <p>– <b>CHANGED:</b> In Sec. 5, added new row “diversity level” in Table 2, including examples “vendors” as an example.</p>	R71

#	Ref	Old location	F: Comments by Tanja Lange (TUE)	Related	Reply Notes	Rev
92	F1		I was surprised that the draft is a meta document and apart from the basic Blakey and Shamir schemes does not include any details. The questions posted are reasonable and relevant when evaluating schemes. I understand that making selections will make the document much more controversial but I think it would also make it more useful. Covering everything from basic secret sharing schemes to applications of MPC in side-channel protection is a daunting task and I wonder what the future of this document is meant to be – are there going to be many appendices covering the actual schemes, organized by application or by technology used?	<a href="#">D1</a> , <a href="#">E1</a> , <a href="#">F1</a> , <a href="#">G1</a> , <a href="#">H2</a> , <a href="#">H8</a> , <a href="#">I1</a> , <a href="#">J1</a> , <a href="#">K1</a> , <a href="#">L1</a> , <a href="#">M2</a>	– <b>NOTE:</b> The <a href="#">last paragraph</a> of <a href="#">Sec. 1</a> explains our decision to not cover the most recent research results. This document intends to motivate an initial engagement with/from stakeholders. We hope to hear from stakeholders, e.g., during NTCW’19, about comparative advantages of various threshold schemes. We do not envision updating this document with appendices, but future documents in the scope of this project may cover more concrete technical material. – <b>CHANGED:</b> No change.	—
93	F2		Editorial comment: It is a bit confusing that intro and section 2 talk about f-out-of-n threshold schemes and section 1 talks about k-out-of-n secret sharing schemes. A short version of section 5.1 given in the introduction would help to clarify that there are two different directions. It might also help to highlight this in the examples, so mention 1-out-of-2 threshold or basic Blakey etc. I agree that both are common but this can be confusing.	<a href="#">A2</a> , <a href="#">E45</a> , <a href="#">E60</a> , <a href="#">F2</a> , <a href="#">K11</a> , <a href="#">K12</a>	– <b>NOTE:</b> Agree with needed clarifications. – <b>CHANGED:</b> See reply to <a href="#">A2</a> .	<a href="#">R3</a> , <a href="#">R13</a> , <a href="#">R17</a> , <a href="#">R18</a> , <a href="#">R49</a> , <a href="#">R74</a> , <a href="#">R72</a>
94	F3	Section 2.2	Content comment: I think it’s a missed chance to not mention that resharing for Shamir works without ever reconstructing the secret, so also without the need for a dealer. I would like to see a short section on this after line 418. Ensuring that devices securely forget data is a problem in practice and an important reason for using secret-sharing schemes is the requirement not to trust a single party, so there should never be a party that knows the secret – for generation or for resharing or for usage. If you would rather include a reference than a worked out section, may I suggest my work with Yvo Desmedt <a href="https://link.springer.com/chapter/10.1007%2F11889663_12">https://link.springer.com/chapter/10.1007%2F11889663_12</a> in which we go through the full details of what this means on the example of ID-based cryptography.	<a href="#">E54</a> , <a href="#">F3</a>	– <b>NOTE:</b> Resharing without a dealer is a relevant capability. – <b>CHANGED:</b> Added a short paragraph in <a href="#">Sec. 3.1.3</a> , mentioning the possibility of resharing without a dealer.	<a href="#">R44</a>

#	Ref	Old location	F: Comments by Tanja Lange (TUE)	Related	Reply Notes	Rev
95	F4	Section 3.1	Content comment: The example of RSA signatures is nice in that it shows both the applicability and the restrictions. However, it would also be good to include a fully positive example, such as ElGamal decryption or DH key share computation where group order is known and thus $k$ -out-of- $n$ schemes can be used in a distributed manner (as opposed to $n$ -out-of- $n$ ).	<a href="#">E34</a> , <a href="#">E35</a> , <a href="#">E39</a> , <a href="#">F4</a>	<p>– <b>NOTE:</b> In old line 549 there was already a citation that achieves <math>k</math>-out-of-<math>n</math> even for an RSA-based scheme. We appreciate that the use of groups of known order can sometimes facilitate some properties in threshold schemes (e.g., as mentioned in old lines 568-572). Old lines 570–579 also alluded to easier operations when the group order is known.</p> <p>– <b>CHANGED:</b> We intend to avoid intricate details, but, to emphasize the suggested point, made some edits in the “<math>k</math>-out-of-<math>n</math> threshold scheme” <a href="#">paragraph</a>: mentioned the aspect of knowledge vs. lack-of-knowledge of group order; cited <a href="#">[DF90]</a> as complementary example; also new citation <a href="#">[GJKR99]</a> in Sec. 3.1.3.</p>	<a href="#">R38</a>

#	Ref	Old location	G: Comments by Yehuda Lindell (Bar-Ilan and Unboundtech)	Related	Reply Notes	Rev
96	G1		We would like to begin by strongly supporting this initiative by NIST to standardize threshold schemes for cryptographic primitives.	<a href="#">D1</a> , <a href="#">E1</a> , <a href="#">F1</a> , <a href="#">G1</a> , <a href="#">H2</a> , <a href="#">H8</a> , <a href="#">I1</a> , <a href="#">J1</a> , <a href="#">K1</a> , <a href="#">L1</a> , <a href="#">M2</a>	<p>– <b>NOTE:</b> Thank you for the encouragement.</p> <p>– <b>CHANGED:</b> No change.</p>	—
97	G2		Secure multiparty computation (MPC) has reached a level of maturity that makes it suitable for solving the problem of key protection via key distribution, without the key ever being reassembled. There are fast protocols for all of the standard cryptographic algorithms (RSA key generation, signing and decryption, DH/ECDH, IES/ECIES, ECDSA/DSA, AES with all modes of operation, HMAC and so on), and there is considerable interest from industry in deploying these solutions. As such, we now see a number of startups and larger companies deploying MPC for this purpose. We have three main comments, that we outline below.	<p>Industry: <a href="#">B1</a>, <a href="#">G2</a>, <a href="#">I2</a>, <a href="#">M1</a>;</p> <p>Schemes: <a href="#">E25</a>, <a href="#">E27</a>, <a href="#">G2</a>.</p>	<p>– <b>NOTE:</b> This NISTIR intends to promote discussion, including with industry, about threshold schemes for cryptographic primitives.</p> <p>– <b>CHANGED:</b> No change.</p>	—
98	G3		<b>FIPS certification.</b> The purpose of standardization of protocols in this area must be to enable FIPS certification at a higher level than currently now possible. Otherwise, it is not clear what concrete advantage one would obtain by having these standards. It is currently possible to achieve FIPS 140-2 level 1 and level 2 using MPC and threshold schemes (by having the different modules connected via FIPS 140-2 certified TLS channels), where level 2 additionally requires certified OS and hardware. However, FIPS 140-2 levels 3 and 4 are currently reserved for dedicated HSMs only.	<a href="#">G3</a> , <a href="#">I7</a> , <a href="#">M6</a>	<p>– <b>NOTE:</b> The aspect of validation/certification is important, and the development of new standards and the various nuances of characterizing features may involve reconsidering validation requirements. In old lines 1477–1478, in Sec. 8, we mention that the “fit into FIPS 140-2 ... is a matter for discussion”. Old representative question #18 (new #6a) asks to check whether security assertions match FIPS 140.</p> <p>– <b>CHANGED:</b> In Sec. 6.2, added new <a href="#">paragraph</a> mentioning that “the process towards standardization of threshold schemes may involve reconsidering the adequacy of the validation requirements and where necessary devise new of complementary requirements”.</p>	<a href="#">R89</a>

#	Ref	Old location	G: Comments by Yehuda Lindell (Bar-Ilan and Unboundtech)	Related	Reply Notes	Rev
99	G4		We believe that the threat model achieved via MPC and threshold cryptography is different but comparable to that of dedicated HSMs, with each having advantages and disadvantages. For example, MPC solutions can be compromised if a quorum of participants are breached, unlike HSMs. However, any flaw inside an HSM is extremely expensive and difficult to recover from, increasing the expected loss. Due to the inflexibility of hardware, this also means that HSMs often do not support the best latest standards and take a long time to deploy new standards. In addition, the main threat that is considered in levels 3 and 4 beyond lower levels is due to attacks requiring physical access. In today's cloud environments, such physical attacks can be greatly mitigated using other means. Finally, in the modern computing era with cloud computing, HSMs in the cloud lose a lot of their security benefits (primarily since they are no longer under the organization's physical control). As such, one should have alternatives in threshold schemes and MPC that provide level 3 and 4 security, and can be deployed in such scenarios.	G4, E14–E16.	<p>– <b>NOTE:</b> There are relevant differences between HSMs vs. cloud nodes.</p> <p>– <b>CHANGED:</b> See reply to HSM-related items E14–E16.</p>	R7
100	G5		<b>What to standardize.</b> It appears to be natural to standardize popular tools, like Shamir secret Sharing, Yao's garbled circuits, oblivious transfer, and so on. We support standardizing these, but warn against these being the only thing certified. This is due to the fact that constructing a secure MPC protocol involves far more than the use of good primitives. It is true that certified encryption schemes can also be misused, but with MPC it is much harder to do this correctly.	G5, I6, J7, M3, M4	<p>– <b>NOTE:</b> Composability is an important matter to consider.</p> <p>– <b>CHANGED:</b> In Sec. 7.2, added a new paragraph highlighting the caution needed with composability of modular components. The new text in Sec. 4.1.4 mentions composability as one aspect to consider in proofs. The new representative question 5e asks about composability.</p>	R100, R108

#	Ref	Old location	G: Comments by Yehuda Lindell (Bar-Ilan and Unboundtech)	Related	Reply Notes	Rev
101	G6		To make this situation worse, the rate of advancement of MPC is so great that any particular protocol that is standardized will most likely be out of date very quickly. As such, we propose that a specific suite of protocols for all of the standard cryptographic algorithms be standardized (including RSA, DH/ECDH, DSA/ECDSA, and symmetric algorithms), but it be explicitly stated that improvements on these protocols also be allowed. This raises a problem as to how a different protocol can be certified. We argue that publication in a recognized journal for peer review (like the Journal of Cryptology) and/or external validation via a recognized MPC researcher at a recognized institution, should be valid. We are fully aware that defining such requirements is a huge minefield. As such, we don't have an answer as to how this should be done, but we do strongly argue that a way must be found. Otherwise, the standardization process may yield little benefit.	G6, J5	<p>– <b>NOTE:</b> The process of standardizing new threshold schemes will consider the “NIST cryptographic standards and guidelines development process” (NISTIR 7977) [Gro16]. Useful experience will be gained throughout the process of standardization.</p> <p>– <b>CHANGED:</b> In Sec. 4.1.4 (“proofs of security”) added a reference to the “NIST Cryptographic Standards and Guidelines Development Process” (NISTIR 7977) [Gro16], which establishes principles that NIST follows for the development of crypto standards.</p>	R59
102	G7		<b>Proofs of security.</b> In general, proofs of security are now well accepted as necessary for cryptographic schemes. For MPC, they are absolutely essential. We argue that both simulation-based and game-based definitions should be acceptable, but that security should always be required in the presence of malicious adversaries with a reasonable security parameter (e.g., computational parameter at the level of the algorithm, and statistical parameter at least $2^{-40}$ ). There are some cases where weaker levels of security in the presence of malicious adversaries, like that provided by dual execution, can also be considered. These issues need to be clarified and discussed as the tradeoffs are important.	G7, I38, J8, J9, J10, H6	<p>– <b>NOTE:</b> We agree that proofs are nowadays essential.</p> <p>– <b>CHANGED:</b> In Sec. 4.1.4 (“proofs of security”), mentioned several aspects to consider in proofs, including types of proof and security parameters. In Sec. 7.1, replaced old question #14 with more focused questions (5e, 5f, 5g) that ask about properties of the proof of security.</p>	R58, R100, R101, R102

#	Ref	Old location	H: Comments by Samuel Ranellucci (Unboundtech)	Related	Reply Notes	Rev
103	H1		To summarize our comments, we believe that having different types of standards would result in simpler standards that are easier to understand.	H1, H3, H9	– <b>NOTE:</b> See reply to item H3 – <b>CHANGED:</b> No change.	—
104	H2		We would like to thank NIST for taking the initiative on standardizing threshold implementations.	D1, E1, F1, G1, H2, H8, I1, J1, K1, L1, M2	– <b>NOTE:</b> Thank you for the encouragement. – <b>CHANGED:</b> No change.	—
105	H3		We believe that NIST could make it easier to standardize threshold implementations by creating different types of specifications. We begin by describing a problem that could be solved by creating different types of specifications.	H1, H3, H9	– <b>NOTE:</b> The aspect of granularity of standards (Sec. 7.2) is a main open question. – <b>CHANGED:</b> No change.	—
106	H4		To prove a threshold implementation secure, it is necessary to define many features. The list of features includes but is not limited to the necessity of a trusted setup, access to some hybrid functionalities, access to cryptographic primitives, the existence of private and authenticated channels between parties, and includes the power of the adversary (semi-honest or malicious, static or adaptive). Already, these issues would make constructing a specification complex. However, if we also include issues such as “What levels of diversity are envisioned to deter common-mode failures?” and “Is the system model applicable to known and relevant application contexts?”, this would make standardization even more complicated. To remedy this problem, we recommend three types of specifications: primitive standardization, protocol standardization and application standardization.		– <b>NOTE:</b> The report enumerates several of these many features. Levels of diversity are now further exemplified in Table 2. Composability is an important consideration when considering composition of different layers of specifications. – <b>CHANGED:</b> No change.	—
107	H5		<b>Primitive Specifications:</b> The first type of specifications are implementations of cryptographic primitives in the same vein as AES and SHA2 standardization. Standardizing garbling schemes and secret sharing fall under this category.	H5, I31, J6	– <b>NOTE:</b> Relates to representative question 2c. The aspect of granularity of standards (Sec. 7.2) is a main open question. – <b>CHANGED:</b> New paragraph in Sec. 7.2 about modular components. See also the reply to items about composability (G5, I6, J7, M3, M4).	R108

#	Ref	Old location	H: Comments by Samuel Ranellucci (Unboundtech)	Related	Reply Notes	Rev
108	H6		<b>Protocol Specifications:</b> The second type of specifications would be directed at interactive protocols. For these specifications, the model should not contain system information and should only contain the model typically found in cryptographic papers. In addition, I would recommend that unless stated otherwise, these specifications should assume private and authenticated channels, static security and malicious adversaries since this is the most relevant model in the literature. GMW-based protocols and garbled-circuit protocols fall under this category.	<a href="#">H6</a> , <a href="#">K8</a> (Channels); <a href="#">G7</a> , <a href="#">H6</a> , <a href="#">I7</a> , <a href="#">L2</a> , <a href="#">L3</a> (types of adversary).	<p>– <b>NOTE:</b> We intend that the properties (requirements or possible restrictions) of communication channels and types of adversaries be specified explicitly. Static and active security adversaries constitute a very relevant class, among several cases mentioned in Sec. 4.2. Types of channels may vary considerably with the platform, and between inter-node and node-to-client types. The question of which cases are more relevant is open for consideration for new standards.</p> <p>– <b>CHANGED:</b> In Sec. 4.1, the new <a href="#">paragraph</a> “proofs of security” mentions types of adversary as an aspect to consider.</p>	<a href="#">R60</a>
109	H7		<b>Deployment Specifications:</b> Finally, in the last type of specification, the specifications focus on system issues that are crucial to the deployment of threshold implementations. These specifications, when built upon the second type specifications, could be described at a higher level of abstraction. Threshold implementations of private-data storage and distributed password verification would fall under this category. We are aware that for certain applications, precise deployment specifications could not be generated due to the variance of requirements. However, it may be the case that for certain applications, standardization would be beneficial.	<a href="#">C1</a> , <a href="#">I2</a> , <a href="#">I5</a> , <a href="#">E30</a> , <a href="#">H7</a> (higher-level apps)	<p>– <b>NOTE:</b> See item <a href="#">I5</a>.</p> <p>– <b>CHANGED:</b> See item <a href="#">I5</a>.</p>	<a href="#">R12</a>
110	H8		In conclusion, we thank NIST for their effort.	<a href="#">D1</a> , <a href="#">E1</a> , <a href="#">F1</a> , <a href="#">G1</a> , <a href="#">H2</a> , <a href="#">H8</a> , <a href="#">I1</a> , <a href="#">J1</a> , <a href="#">K1</a> , <a href="#">L1</a> , <a href="#">M2</a>	<p>– <b>NOTE:</b> See reply to <a href="#">H2</a>.</p> <p>– <b>CHANGED:</b> No change.</p>	—
111	H9		We believe that our recommendation of having different types of standards would result in simpler standards that are easier to understand.	<a href="#">H1</a> , <a href="#">H3</a> , <a href="#">H9</a>	<p>– <b>NOTE:</b> See reply to item <a href="#">H3</a></p> <p>– <b>CHANGED:</b> No change.</p>	—

#	Ref	Old location	I: Comments by Dan Bogdanov (Cybernetica)	Related	Reply Notes	Rev
112	I1		Cybernetica is happy to see that threshold cryptography is gaining momentum and best practices are being collected by NIST. We see this as a critical step for the new technologies to be accepted in the United States of America.	D1, E1, F1, G1, H2, H8, I1, J1, K1, L1, M2	<p>– <b>NOTE:</b> Thank you for the encouragement.</p> <p>– <b>CHANGED:</b> No change.</p>	—
113	I2		We have multiple US partners who are evaluating secure multi-party computation for data analytics uses.	B1, G2, I2, M1(industry), C1, I2, I5, E30, H7 (higher-level apps)	<p>– <b>NOTE:</b> This NISTIR intends to promote discussion, with multiple stakeholders, about threshold schemes for cryptographic. SMPC is a main approach to achieve threshold schemes.</p> <p>– <b>CHANGED:</b> No change, but see item I5.</p>	—
114	I3		This document consists of suggestions to further development in this field and responses to a subset of questions. Please find the responses to questions in separate chapters		<p>– <b>NOTE:</b> We appreciate the comments/answers provided. Many of these questions are intended to be addressed in the context of concrete schemes, i.e., what should be questioned/considered when analyzing concrete schemes, or when pondering about how to ask for proposals for new schemes.</p> <p>– <b>CHANGED:</b> No change.</p>	—
115	I4		<b>Suggestion S1:</b> If the aim is to set the scope on using threshold cryptography for cryptographic algorithms (e.g., key generation, encryption, decryption, signatures), then be as clear as possible about this. Threshold cryptography is a general-purpose technique and can be used for big data, statistics etc applications. If NIST standardises the use of these techniques for encryption and signature use, make sure that the same restrictions may not be reasonable for data analytics use.	I5	<p>– <b>NOTE:</b> See reply to item I5.</p> <p>– <b>CHANGED:</b> No change.</p>	—
116	I5		<b>Suggestion S2:</b> If the aim is to set the scope on using threshold cryptography for a wide range of applications (e.g., key generation, encryption, decryption, signatures, but also data general purpose processing), then allow these applications to have separate characteristics.	C1, I2, I5, E30, H7 (higher-level apps)	<p>– <b>NOTE:</b> This report is focused on threshold schemes for “cryptographic primitives”. There are higher-level applications that can take advantage of such schemes.</p> <p>– <b>CHANGED:</b> Added a brief note on the motivation for higher-level applications, in a place that is immediately followed by a paragraph that conveys the focus of this report: cryptographic primitives.</p>	R12
117	I6		<b>Suggestion S3:</b> If the scope will be on wider applications, avoid standardizing monolithic protocols. Support and encourage composition of primitives.	G5, I6, J7, M3, M4	<p>– <b>NOTE:</b> See reply to item I5.</p> <p>– <b>CHANGED:</b> About composability, see reply to G5.</p>	R108

#	Ref	Old location	I: Comments by Dan Bogdanov (Cybernetica)	Related	Reply Notes	Rev
118	I7		<b>Suggestion S4:</b> Avoid limitations to certain security levels. E.g., some applications may be provably secure in the honest-but-curious security model, but will protect against other attacks using other approaches.	<a href="#">G3</a> , <a href="#">H6</a> , <a href="#">I7</a> , <a href="#">L2</a> , <a href="#">L3</a>	<p>– <b>NOTE:</b> Evaluation of potential threshold schemes and their relations with security levels is a relevant matter.</p> <p>– <b>CHANGED:</b> See replies to <a href="#">G3</a> (on security levels) and <a href="#">L2</a> (on different types of adversary).</p>	<a href="#">R66</a> , <a href="#">R89</a>
119	I8		<b>Suggestion S5:</b> Encourage machine-generated and machine-checked protocol implementations.	<a href="#">E11</a> , <a href="#">I8</a>	<p>– <b>NOTE:</b> Relates to automated validation.</p> <p>– <b>CHANGED:</b> See item <a href="#">E11</a>.</p>	<a href="#">R91</a> , <a href="#">R96</a>
120	I9	m	<p>Section 5.1, lines 1040-1041 3 QUESTIONS ON THRESHOLD VALUES 3.1 QUESTION 1: FOR THE DESIRED SECURITY PROPERTIES, WHAT ARE THE THRESHOLD VALUES (<math>F</math> AND/OR <math>K</math>), AS A FUNCTION OF THE TOTAL NUMBER <math>N</math> OF COMPONENTS?</p> <p>It is not clear if this question assumes that all parties in the system (including customers) are involved in <math>n</math>. If yes, then the threshold may be a function of <math>n</math>. But if some users of the system are not part of <math>n</math> (e.g., they delegate their computation to the parties holding the shares), then it does not have to be a function. It may take a few shares per input (2, 3 even) to process inputs from millions of users. This highly depends on the application, as raising the number of components often reduces performance. We recommend no hard restrictions here.</p>		<p>– <b>NOTE:</b> When the threshold system is conceived as cryptographic module (to perform some cryptographic primitive), the “customers” (users, requesters of cryptographic operations) are typically not included in <math>n</math>. More elaborate answers may vary with the system model.</p> <p>– <b>CHANGED:</b> No change.</p>	—
121	I10	Section 5.1, lines 1042-1043	<p>3.2 QUESTION 2: WHAT ENVISIONED APPLICATION CONTEXTS JUSTIFY A HIGH THRESHOLD FOR SOME PROPERTIES AT THE COST OF A LOW THRESHOLD FOR OTHER PROPERTIES (OR OF OTHER MITIGATION MEASURES)?</p> <p>In a way, the threshold gives us a slider between control-confidentiality and integrity-reliability. Low thresholds ensure that even if some parties misbehave by sending wrong results or not sending anything at all, the results can still be calculated. High thresholds are better for confidentiality. We recommend no hard restrictions here.</p>		<p>– <b>NOTE:</b> There are various possible tradeoffs.</p> <p>– <b>CHANGED:</b> No change.</p>	—

#	Ref	Old location	I: Comments by Dan Bogdanov (Cybernetica)	Related	Reply Notes	Rev
122	I11	Section 5.1, lines 1044-1045	<p>3.3 QUESTION 3: HOW DO THRESHOLD VALUES VARY WITH RESPECT TO CONCEIVABLE VARIATIONS OF THE SYSTEM MODEL (E.G., SYNCHRONY VS. ASYNCHRONY, PASSIVE VS. ACTIVE ADVERSARIES)?</p> <p>There are well-known possibility results, several of them cited in Draft NISTIR 8214 (CCD87 etc). However, we are not aware of a function that takes as input the system model and outputs a recommended number of parties. We see that the number of components and thresholds are often application-specific. We recommend no hard restrictions here.</p>	E55, I11	<p>– <b>NOTE:</b> Thresholds may vary with the system model.</p> <p>– <b>CHANGED:</b> No direct change but [CCD88] was added.</p>	R32
123	I12	Section 5.2, line 1079	<p>4 QUESTIONS ON COMMUNICATION INTERFACES 4.1 QUESTION 1: ARE CLIENTS AWARE OF THE THRESHOLD NATURE OF THE IMPLEMENTATION?</p> <p>If the human users are not, then the client interfaces (libraries, websites, mobile apps, adapters or whatever) have to be. Pushing the actual secret sharing process to the client gives better auditability and security in all constructions that we are aware of. We recommend no hard restrictions here.</p>	E69, I12	<p>– <b>NOTE:</b> Auditability may be a feature of certain threshold schemes, e.g., for multi-signatures, as mentioned in Sec. 3.1.4. For certain applications there may be advantages in having the threshold interface not reveal the threshold internals. See complementary observation in item E69.</p> <p>– <b>CHANGED:</b> No change.</p>	—
124	I13	Section 5.2, line 1080	<p>4.2 QUESTION 2: HOW IS THE INITIAL REQUEST FROM A CLIENT PROPAGATED THROUGH THE SET OF NODES?</p> <p>This is highly application-specific. In some cases, encrypting shares at the customers and propagating them to the shareholders through a single gateway is preferable, sometimes it is better to send shares directly to the parties that will process them. We recommend no hard restrictions here.</p>		<p>– <b>NOTE:</b> The preference can depend on the application and also on the attack model.</p> <p>– <b>CHANGED:</b> No change.</p>	—
125	I14	Section 5.2, line 1081	<p>4.3 QUESTION 3: HOW CAN THE INTER-NODE COMMUNICATION BE COMPROMISED?</p> <p>Given that threshold cryptography implementations are often modern, they have a better probability to use up-to-date secure channel techniques. In practice, we see that compromises are more probable at the endpoints.</p>	E65, I14, K10	<p>– <b>NOTE:</b> Communication channels may vary substantially, e.g., with the implementation platform.</p> <p>– <b>CHANGED:</b> No change. Added more examples to the corresponding row in Table 2.</p>	R68

#	Ref	Old location	I: Comments by Dan Bogdanov (Cybernetica)	Related	Reply Notes	Rev
126	I15	Section 5.2, lines 1082-1083	<p>4.4 QUESTION 4: HOW DOES THE CLIENT OBTAIN A CONSOLIDATED REPLY BASED ON A SET OF PARTIAL RESULTS PRODUCED BY A SET OF NODES?</p> <p>This is application-specific. However, in most applications, we see that the client often has the best information to decide on the results. This is because the client can receive partial results from various committees and several nodes.</p>		<p>– <b>NOTE:</b> It may depend on the application.</p> <p>– <b>CHANGED:</b> No change.</p>	—
127	I16	Section 5.2, lines 1084-1085	<p>4.5 QUESTION 5: HOW IS THE LOGICAL/PHYSICAL “BOUNDARY” (SEE FIPS 140-2 [NIS18C]) OF THE SYSTEM AFFECTED BY THE EXISTING COMMUNICATION CHANNELS?</p> <p>This is application-specific. In some settings, we consider a set of parties holding the shares for a single value a single logical computer. E.g., in a general-purpose computation scenario based on secure multi-party computation, all parties holding the shares form a single application server.</p>		<p>– <b>NOTE:</b> For validation purposes, answers will also depend on implementation details, prior to deployment for a higher-level application.</p> <p>– <b>CHANGED:</b> No change.</p>	—
128	I17	Section 5.3, lines 1200-1201	<p>5 QUESTIONS ON TARGET COMPUTING PLATFORMS</p> <p>5.1 QUESTION 1: IF A PROPOSED THRESHOLD SCHEME IS DEvised FOR A “SINGLE-DEVICE” SETTING, WHAT CAN GO WRONG IF ITS COMPONENTS ARE INSTEAD SEPARATED AND COMMUNICATE OVER THE INTERNET?</p> <p>This question does not have enough information for a certain response. One could design a scheme that works in a single-device setting and also remains secure in a distributed setting. And, in the case of a protocol based on trusted execution environments, this could not work.</p>		<p>– <b>NOTE:</b> The answer may vary, e.g., depending on the composability guarantees of the communication “module”.</p> <p>– <b>CHANGED:</b> No change.</p>	—
129	I18	Section 5.3, lines 1202-1203	<p>5.2 QUESTION 2: WHICH PARTS OF THE LOGICAL BOUNDARY OF THE THRESHOLD SYSTEM DO NOT CORRESPOND TO A PHYSICAL BOUNDARY, AS VERIFIED BY THE SYSTEM DEVELOPER OR DEPLOYER?</p> <p>No response at this time.</p>		—	—

#	Ref	Old location	I: Comments by Dan Bogdanov (Cybernetica)	Related	Reply Notes	Rev
130	I19	Section 5.3, lines 1204-1205	<p>5.3 QUESTION 3: IS THE SYSTEM SIMPLY DEVELOPED AT THE SOFTWARE LAYER, OR ARE THERE SOFTWARE COMPONENTS TIED TO PARTICULAR HARDWARE COMPONENTS?</p> <p>We can imagine both kinds of solutions. Consider threshold protocols where it is important that a certain part of the deployment is in a mobile phone or in a Trusted Execution Environment. We recommend no hard restrictions here.</p>	E17, I19	<p>– <b>NOTE:</b> We can also imagine hybrid solutions.</p> <p>– <b>CHANGED:</b> No change.</p>	—
131	I20	Section 5.3, lines 1206-1207	<p>5.4 QUESTION 4: WHICH AUXILIARY COMPONENTS SUPPORT THE THRESHOLD SCHEME BUT HAVE A FAILURE MODEL DIFFERENT FROM THE ONE APPLIED TO THE THRESHOLD NODES?</p> <p>No response at this time.</p>		—	—
132	I21	Section 5.4, lines 1299	<p>6 QUESTIONS ON SETUP AND MAINTENANCE 6.1 QUESTION 1: CAN A THRESHOLD SCHEME BE BOOTSTRAPPED IN BOTH DEALER AND DEALER-FREE MANNERS?</p> <p>We can imagine various deployments. In dealer-free environments, shares could be generated by parties. In single-dealer environments, there will be one authoritative party for shares. In a multi-dealer environment, there will be multiple dealers who all know the scheme and the parties who to share the secrets to. Thus, the answer would be yes, and we also foresee many multi-dealer deployments.</p>		<p>– <b>NOTE:</b> We can also envision multiple solutions. An open question is whether a single standard for a threshold scheme should/may incorporate both types of solutions. The multi-dealer observation scenario is interesting.</p> <p>– <b>CHANGED:</b> No change.</p>	—
133	I22	Section 5.4, lines 1300	<p>6.2 QUESTION 2: WHAT LEVELS OF DIVERSITY ARE ENVISIONED TO DETER COMMON-MODE FAILURES?</p> <p>No response at this time.</p>		—	—
134	I23	Section 5.4, lines 1301	<p>6.3 QUESTION 3: WHAT DEPENDENCY OF COMPROMISE EXISTS ACROSS NODES, FOR ENVISIONED ATTACK VECTORS?</p> <p>No response at this time.</p>		—	—
135	I24	Section 5.4, lines 1302	<p>6.4 QUESTION 4: DOES THE SUBPROTOCOL FOR HANDLING REJUVENATIONS INTERFERE WITH THE SYSTEM AVAILABILITY?</p> <p>No response at this time.</p>		—	—

#	Ref	Old location	I: Comments by Dan Bogdanov (Cybernetica)	Related	Reply Notes	Rev
136	I25	Section 7, lines 1374-1377	7 QUESTIONS ON CRITERIA FOR STANDARDIZATION FOR NEW SCHEMES 7.1 QUESTION 1: ARE THE CHARACTERIZING FEATURES OF THE THRESHOLD SCHEME FULLY DESCRIBED? We believe that once the scope for standardization is set (see Suggestions S1 and S2), it will become easier to figure out the required characteristics.		– <b>NOTE:</b> Agreed; here we also wanted to propose that once a new scheme is analyzed one should ask if/which of its characterizing features are well described. – <b>CHANGED:</b> No change.	—
137	I26	Section 7, lines 1378	7.2 QUESTION 2: ON WHAT EXECUTING PLATFORMS CAN THE SCHEME BE IMPLEMENTED? A relevant question, if there are restrictions (e.g., Trusted Execution Environments like ARM TrustZone or Intel SGX).		– <b>NOTE:</b> Different schemes may require different assumptions on trusted components, so some schemes/assumptions might not be possible to implement/cover in some platforms. – <b>CHANGED:</b> No change.	—
138	I27	Section 7, lines 1379	7.3 QUESTION 3: WHAT ARE THE OPERATIONAL COSTS AND PROPERTIES OF SETUP AND MAINTENANCE? Often this will be dependent on third party services, e.g. clouds.		– <b>NOTE:</b> It will be pertinent for a new threshold scheme proposal to estimate related costs. – <b>CHANGED:</b> No change.	—
139	I28	Section 7, lines 1380	7.4 QUESTION 4: WHAT ARE THE NODE-REJUVENATION MECHANISMS (E.G., RESHARING OR NODE REPLACEMENT)? A relevant question.		– <b>NOTE:</b> Agreed. – <b>CHANGED:</b> No change.	—
140	I29	Section 7, lines 1381	7.5 QUESTION 5: HOW EFFICIENT/PERFORMANT ARE THE OPERATIONS AS A FUNCTION OF THRESHOLD PARAMETERS? A relevant question. But note that some schemes might have a fixed n and k, still being threshold schemes.		– <b>NOTE:</b> Even for fixed parameters, it will be relevant to compare costs vs. the conventional (non-threshold) system. When parameters can vary, it will also be interesting to know how the cost varies. – <b>CHANGED:</b> No change.	—
141	I30	Section 7, lines 1382	7.6 QUESTION 6: IS THE SCHEME APPLICABLE TO NIST-APPROVED CRYPTOGRAPHIC PRIMITIVES? A relevant question.		– <b>NOTE:</b> Agreed. – <b>CHANGED:</b> No change.	—
142	I31	Section 7, lines 1383	7.7 QUESTION 7: DO BASE PRIMITIVES (E.G., OBLIVIOUS TRANSFER) REQUIRE INDEPENDENT STANDARDIZATION? A relevant question. However note that this should be informative, as the use of a new useful protocol should not be delayed because a primitive is not standardized.	H5, I31, J6	– <b>NOTE:</b> This may become relevant if it provides advantages for standardization of higher-level primitives. For now it is an open question. – <b>CHANGED:</b> No change. See reply to item H5.	—

#	Ref	Old location	I: Comments by Dan Bogdanov (Cybernetica)	Related	Reply Notes	Rev
143	I32	Section 7, lines 1384	7.8 QUESTION 8: IS THE SYSTEM MODEL APPLICABLE TO KNOWN AND RELEVANT APPLICATION CONTEXTS? A relevant question.		– <b>NOTE:</b> Agreed. – <b>CHANGED:</b> No change.	—
144	I33	Section 7, lines 1385	7.9 QUESTION 9: HOW IS DIVERSITY OF NODES RELATED TO KNOWN ATTACK VECTORS? A relevant question. The deployment of a system might be more secure if its nodes are deployed on different hardware-software combinations.		– <b>NOTE:</b> Diversity is relevant for threshold schemes. – <b>CHANGED:</b> We have added a new line to Table 2, for “diversity levels”	—
145	I34	Section 7, lines 1386	7.10 QUESTION 10: IS THE IMPLEMENTATION COMPLEXITY LIKELY TO LEAD TO NEW BUGS OR MISCONFIGURATION? That is not a useful question. An honest answer will always be "yes", but this should not stop useful components from being built.		– <b>NOTE:</b> The goal of the question is to promote caution on possible new bugs and misconfigurations that may arise from the threshold approach. Somewhat related, Sec. 4.2 also considers “threshold-related” attacks. – <b>CHANGED:</b> Edited the question to become more open, asking to identify which aspects can lead to new bugs or misconfiguration	R97
146	I35	Section 7, lines 1387	7.11 QUESTION 11: WHAT TRUSTED SETUP AND ASSUMPTIONS ARE REQUIRED (E.G., DEALER, SPECIAL COMPONENTS)? A most relevant question.		– <b>NOTE:</b> Agreed. – <b>CHANGED:</b> No change.	—
147	I36	Section 7, lines 1388	7.12 QUESTION 12: WHAT THRESHOLD PROPERTIES RELATE TO RESISTANCE AGAINST SIDE-CHANNEL ATTACKS AND HOW? A question with uncertain relevance. Resistance against side-channel attacks might come from other aspects of the system.		– <b>NOTE:</b> Side-channels resistance can relate to different parts of a system. Some threshold approaches may mitigate some side-channel attacks. – <b>CHANGED:</b> No change.	—
148	I37	Section 7, lines 1389	7.13 QUESTION 13: ARE THERE IDENTIFIED SECURITY TRADEOFFS ACROSS ATTACK TYPES AND CONFIGURATIONS? A very relevant question.		– <b>NOTE:</b> Agreed. – <b>CHANGED:</b> No change.	—
149	I38	Section 7, lines 1390	7.14 QUESTION 14: IS THE SECURITY ASSESSMENT SUPPORTED BY A SECURITY PROOF ? A very relevant question.	G7, I38, J8, J9, J10	– <b>NOTE:</b> Agreed. – <b>CHANGED:</b> See the reply to G7.	R58, R100, R101, R102

#	Ref	Old location	I: Comments by Dan Bogdanov (Cybernetica)	Related	Reply Notes	Rev
150	I39	Section 7, lines 1391	7.15 QUESTION 15: HOW DOES THE RELIABILITY COMPARE AGAINST THAT OF A CONVENTIONAL IMPLEMENTATION? A question with uncertain relevance. If the threshold scheme is built for confidentiality, it will always be less reliable. If it is built for reliability, it will be more reliable.		– <b>NOTE:</b> We mean reliability in upholding a desired security property, e.g., including being reliable in upholding the secrecy of a key. – <b>CHANGED:</b> No change.	—
151	I40	Section 7, lines 1392	7.16 QUESTION 16: HOW BRITTLE IS THE SCHEME (LIKELY TO BREAK UNDER SMALL VARIATIONS IN THE ENVIRONMENT)? A question with uncertain relevance. Isn't this a partial duplicate for questions 9, 10, 11?		– <b>NOTE:</b> Ideally, a good threshold scheme would remain secure under conceivable variations in the environment. – <b>CHANGED:</b> No change.	—
152	I41	Section 7, lines 1393	7.17 QUESTION 17: WHAT FEATURES OF GRACEFUL DEGRADATION EXIST AGAINST CONCEIVABLE FAILURES? A relevant question.		– <b>NOTE:</b> Agreed. – <b>CHANGED:</b> No change.	—
153	I42	Section 7, lines 1394	7.18 QUESTION 18: DO THE SECURITY ASSERTIONS MATCH / FIT INTO THE FIPS 140-2 FRAMEWORK? A relevant question.		– <b>NOTE:</b> Agreed. – <b>CHANGED:</b> No change.	—
154	I43	Section 7, lines 1395	7.19 QUESTION 19: HOW TESTABLE IS THE SCHEME (CAN SECURITY ASSERTIONS BE TESTED AND VALIDATED)? A very relevant question.		– <b>NOTE:</b> Agreed. – <b>CHANGED:</b> No change.	—
155	I44	Section 7, lines 1396	7.20 QUESTION 20: IS THERE A PROPOSED AUTOMATED VALIDATION MECHANISM? A relevant question.		– <b>NOTE:</b> Agreed. – <b>CHANGED:</b> No change.	—
156	I45	Section 7, lines 1397	7.21 QUESTION 21: WHAT ARE THE INTELLECTUAL PROPERTY IMPLICATIONS AND THE LICENSING CONDITIONS? A relevant question.		– <b>NOTE:</b> Agreed. – <b>CHANGED:</b> No change.	—

#	Ref	Old location	J: Comments by Rosario Gennaro (The City College of New York)	Related	Reply Notes	Rev
157	J1		In this short document I present a few comments on the draft NIST document on threshold cryptography. Before I begin, I would like to commend NIST for starting a project of standardization of threshold cryptographic schemes. In the process you are raising the awareness of the security community about the availability of sophisticated cryptographic techniques that can enhance security (intended in the largest possible sense including confidentiality, integrity, availability, etc.).	D1, E1, F1, G1, H2, H8, I1, J1, K1, L1, M2	<p>– <b>NOTE:</b> Thank you for the encouragement.</p> <p>– <b>CHANGED:</b> No change.</p>	—
158	J2		GENERIC VS AD-HOC SOLUTIONS. It should be remarked that <i>any</i> cryptographic primitive can be implemented in a distributed "threshold" manner, via generic secure multiparty computation protocols.	E56, J2	<p>– <b>NOTE:</b> Agreed.</p> <p>– <b>CHANGED:</b> Some adjustments in Sec. 2.4, making this point clear, while also conveying that some threshold techniques can also be conceptualized outside the SMPC framework.</p>	R33, R34
159	J3		The dramatic improvement in speed and reliability of some of those protocols of the last few years suggests that maybe we should consider <ul style="list-style-type: none"> <li>(i) generic solutions for cryptographic primitives for which we do not have ad-hoc protocols;</li> </ul>	J3–J5	<p>– <b>NOTE:</b> Generic solutions are a potential possibility.</p> <p>– <b>CHANGED:</b> See reply to J5.</p>	—
160	J4		<ul style="list-style-type: none"> <li>(ii) a requirement that compares ad-hoc solutions to generic multiparty computation approaches</li> </ul>	J3–J5	<p>– <b>NOTE:</b> Ad-hoc solutions are a potential possibility.</p> <p>– <b>CHANGED:</b> See reply to J5.</p>	—
161	J5		In other words I wonder if this standardization effort should include a standard way to turn any cryptographic primitive into a threshold one using a secure MPC protocol of choice (and standardizing ways to express the primitive into a format that can be fed into the MPC protocol, e.g. how to write it as a circuit, etc.). Once that is done, ad-hoc protocols would have a "standard" benchmark to be compared against the generic approach (and be adopted only if they show substantial improvements over the generic approach).	G6, J3–J5	<p>– <b>NOTE:</b> Might there be synergies to extract between the two types of solutions?</p> <p>– <b>CHANGED:</b> Added related paragraph in Sec. 7.2.</p>	R109

#	Ref	Old location	J: Comments by Rosario Gennaro (The City College of New York)	Related	Reply Notes	Rev
162	J6		For the ad-hoc solutions I also wonder if we need standard protocols for "building blocks" that appear across many solutions. An incomplete list would include: secret sharing, VSS, interpolation in the "exponent" (i.e. compute $g^x$ from shares of $x$ ), computing shares of the inverse of a secret value (i.e. go from a share of $x$ to a share of $x^{-1}$ over some group), and so on.	<a href="#">H5</a> , <a href="#">I31</a> , <a href="#">J6</a>	<p>– <b>NOTE:</b> See reply to item <a href="#">I31</a>.</p> <p>– <b>CHANGED:</b> See reply to item <a href="#">H5</a>.</p>	<a href="#">R108</a>
163	J7		If those building blocks are standardized and proven secure according to a composable definition of security, then the design of protocols for primitives could be greatly simplified, and implementation of different cryptographic primitives would share a common "language".	<a href="#">G5</a> , <a href="#">I6</a> , <a href="#">J7</a> , <a href="#">M3</a> , <a href="#">M4</a>	<p>– <b>NOTE:</b> Composability is an important matter to consider.</p> <p>– <b>CHANGED:</b> See reply to <a href="#">G5</a>.</p>	<a href="#">R108</a>
164	J8		THE NEED FOR SECURITY PROOFS. The draft document asks at the end (question 14) if security proofs should be required. My answer to that question is an emphatic yes. A security proof should guarantee that moving from a centralized to a distributed implementation of the cryptographic primitive would not introduce other weaknesses beyond the ones of the centralized primitive. Therefore if the centralized primitive is assumed secure then the distributed version would be as well, under such a security proof.	<a href="#">G7</a> , <a href="#">I38</a> , <a href="#">J8</a> , <a href="#">J9</a> , <a href="#">J10</a>	<p>– <b>NOTE:</b> Agreed.</p> <p>– <b>CHANGED:</b> See reply to item <a href="#">J9</a>.</p>	—

#	Ref	Old location	J: Comments by Rosario Gennaro (The City College of New York)	Related	Reply Notes	Rev
165	J9		The draft document falls a bit short in my opinion in discussing the options for security proofs. It mentions (somewhat in passing) the real-ideal simulation paradigm which is of course the gold standard for this type of security proofs. However it should be remarked that in some cases proofs based on a "game-based" definitions might suffice. In a game-based definition the salient property of the underlying cryptographic primitive is re-defined in a distributed model where the adversary also has access to the information of $f$ players in the network. For example, if we are distributing a signature scheme, the game-based definition of <i>unforgeability against chosen-message attack</i> can be adapted to the distributed model and the distributed signature proven secure according to this definition. Game-based definitions are weaker since they do not rule out distributed protocols where the adversary might learn information, though they guarantee that this information will not help in breaking the underlying signature. The lack of a generic composability guarantee is the major drawback of this approach.	<a href="#">G7, I38, J8, J9, J10</a>	<p>– <b>NOTE:</b> Agreed.</p> <p>– <b>CHANGED:</b> In Sec. 4.1, added a new paragraph on “Proofs of security”. Also replaced the previous representative question #14 with three new more focused questions (<a href="#">5e</a>, <a href="#">5f</a>, <a href="#">5g</a>) about security proofs.</p>	<a href="#">R58, R100, R101, R102</a>
166	J10		I suggest that the draft document be improved by describing in more detail the possible choices for security proofs, and the various scenarios in which one proof may be acceptable, and that overall the need for security proofs be stressed as mandatory.	<a href="#">G7, I38, J8, J9, J10</a>	<p>– <b>NOTE:</b> Agreed.</p> <p>– <b>CHANGED:</b> See reply to item <a href="#">J9</a></p>	<a href="#">R58</a>
167	J11		THE NEED FOR AVAILABILITY. The document does an excellent job at discussing the trade-offs between confidentiality and availability when using a threshold scheme: if the operation of the scheme requires all $n$ parties to participate, then confidentiality is at its highest, but the scheme will not be robust as a single party may force the system to shut down. On the other hand availability can be trivially improved by replicating the secret keys across $n$ servers, but this makes confidentiality weaker, as now the key is more exposed to attacks. The problem, as discussed in the draft document, is to find the correct "equilibrium point" which may be application dependent.	<a href="#">G4, J11</a>	<p>– <b>NOTE:</b> Availability is an important property of implementations.</p> <p>– <b>CHANGED:</b> See reply to <a href="#">J12</a>.</p>	

#	Ref	Old location	J: Comments by Rosario Gennaro (The City College of New York)	Related	Reply Notes	Rev
168	J12		Availability can always be improved by having a sufficiently high number of servers participate in the protocol, so that even if $f$ servers do not participate there is enough honest servers to maintain functionality of the system. However for efficiency reason one may want to run the protocol with the minimum possible number of servers and hope in an "optimistic" fashion that the servers are honest. But what happens if things go wrong.	<a href="#">E40</a> , <a href="#">J12–J14</a>	– <b>NOTE:</b> See item <a href="#">J14</a> – <b>CHANGED:</b> Added related paragraph in Sec. 4.1.3.	<a href="#">R57</a>
169	J13		One aspect that might be important to discuss in this context, is the ability to detect and reverse faults. For example if $k$ servers are needed to compute the primitive and then the protocol fail because $f$ of those servers might be faulty. The questions here are	<a href="#">J13–J17</a>	– <b>NOTE:</b> Detection of faults and failure is an important matter. – <b>CHANGED:</b> Added representative question <a href="#">4e</a> about fault detection.	<a href="#">R99</a>
170	J14		<ul style="list-style-type: none"> <li>• If <math>k</math> is sufficiently high can we still successfully compute the primitive</li> </ul>	<a href="#">J13–J17</a>	– <b>NOTE:</b> Detectability vs. non-detectability of faults may change the threshold numbers, depending on the type of attack. – <b>CHANGED:</b> Added related paragraph in Sec. 4.1.3.	<a href="#">R57</a>
171	J15		<ul style="list-style-type: none"> <li>• If not, can we at least detect failure (this is easy with signatures as the resulting signature will not verify – but what about other primitives, e.g. PRFs)?</li> </ul>	<a href="#">J13–J17</a> , <a href="#">K6</a>	– <b>NOTE:</b> Different primitives may pose different challenges for correctness verification (e.g., see reply to <a href="#">K6</a> ). – <b>CHANGED:</b> No change.	—
172	J16		<ul style="list-style-type: none"> <li>• If we detect failuer, can we identify the rogue server(s) and remove them from the server pool replacing them with other servers?</li> </ul>	<a href="#">J13–J17</a> , <a href="#">K9</a>	– <b>NOTE:</b> The “removal/replacement” aspect relates to representative question <a href="#">1c</a> . The “identification” part is handled by the reply to <a href="#">J13</a> . – <b>CHANGED:</b> See reply to <a href="#">J13</a> .	
173	J17		These features could be important to have in certain applications.	<a href="#">J13–J17</a>	– <b>NOTE:</b> Applications may benefit from properties of threshold schemes. – <b>CHANGED:</b> No change.	

#	Ref	Old location	K: Comments by Thalia May Laing (HP Inc.)	Related	Reply Notes	Rev
174	K1		We thought the draft document covered some main points related to secret sharing schemes and found it to raise some interesting and important questions. We have the following suggestions as to how the document could be improved and some concepts clarified.	D1, E1, F1, G1, H2, H8, I1, J1, K1, L1, M2	– <b>NOTE:</b> Thank you for the encouragement. – <b>CHANGED:</b> No change.	—
175	K2		(1) Verifiable secret sharing schemes are the main family of schemes recommended to provide robustness and they do so, whilst assuming the dealer is untrusted. Robust secret sharing schemes provide robustness in a more efficient manner whilst assuming the dealer is trusted yet are not recommended as a family of schemes. Is there a requirement for an untrusted dealer? Is there a reason why verifiable secret sharing are suggested and robust schemes are not?	Robustness: K2–K6	– <b>NOTE:</b> See reply to item K5.	—
176	K3		In a verifiable secret sharing scheme (VSS), the dealer is not assumed to be trusted. When a party receives a share from the dealer, they can verify that the share they received is a valid share. Then, when each player submits their share to recover the secret during the reconstruct phase, each share can also be verified to ensure that dishonest players cannot completely disrupt the recovery process. In a robust secret sharing scheme, the dealer is assumed to be trusted and the aim is to prevent dishonest players from corrupting the recovery process by submitting incorrect shares. As we assume the dealer to be trusted we assume the players are initially dealt correct shares.	Robustness: K2–K6	– <b>NOTE:</b> See reply to item K5.	—
177	K4	Lines 128, 1033	Despite this, robust schemes are not suggested as a family of schemes to provide robustness and, instead, verifiable schemes are recommended. For example, in line 128, it is said that 'verifiable secret sharing enables detection of misuse of shares by a shareholder', and on line 1033, verifiable secret sharing schemes are suggested as the solution to a threshold number of parties misbehaving. This is correct, as verifiable secret sharing schemes achieve this, but they also assume an untrusted dealer. Robust schemes do this in a more efficient way (multiple rounds of communication are not necessary) but assuming the dealer to be trusted.	Robustness: K2–K6	– <b>NOTE:</b> See reply to item K5.	—

#	Ref	Old location	K: Comments by Thalia May Laing (HP Inc.)	Related	Reply Notes	Rev
178	K5		It may be helpful to the reader to mention robust schemes as another suggested family of schemes that provide robustness (as well as VSS) and highlight the difference between the trust assumptions on the dealer and the necessary rounds of communication. Examples of robust secret sharing schemes can be found in Section 4 of Krawczyk's 'Secret sharing made short' paper, and in Sections 4 and 5 of Bellare and Rogaway's unified account of secret sharing goals. Krawczyk, H. (1993, August). Secret sharing made short. In Annual International Cryptology Conference (pp. 136-146). Springer, Berlin, Heidelberg. Bellare, M., & Rogaway, P. (2007, October). Robust computational secret sharing and a unified account of classical secret-sharing goals. In Proceedings of the 14th ACM conference on Computer and communications security (pp. 172-184). ACM.	Robustness: <a href="#">K2-K6</a>	<p>– <b>NOTE:</b> Both cases (trusted vs. untrusted dealers) are interesting to consider, as well as the case without dealer.</p> <p>– <b>CHANGED:</b> In Sec. 5.4.1, added mention to the distinction between the two cases: trusted vs. untrusted dealer.</p>	<a href="#">R83</a>
179	K6		(2) It may be useful to the reader to highlight both the efficiency costs and the security requirements on threshold schemes with additional properties (such as robust, verifiable and proactive schemes). Some schemes may have extra communication and complexity costs: - Robust schemes require the communication of more bits from the dealer to the players. - Verifiable secret sharing requires the communication of more bits and multiple rounds of communication. For example, the [AMGC85], a scheme highlighted in the document, is an interactive protocol with multiple rounds of communication. This is an increased communication cost over standard threshold schemes. - In proactive secret sharing schemes, players may have to generate random values and may need to send every other player a value, which is an increased computation and communication cost.	Robustness: <a href="#">J15</a> , <a href="#">K2-K6</a> .	<p>– <b>NOTE:</b> The efficiency of threshold schemes is a relevant matter. As an example, old lines 1029-1031 (new Sec. 5.1.3) mention that verifying the correctness of encryption or decryption may be costlier (see Sec. 5.1.3).</p> <p>– <b>CHANGED:</b> See also the reply to item <a href="#">K5</a>.</p>	<a href="#">R84</a>
180	K7		As well as efficiency costs, the schemes may have additional security requirements:		<p>– <b>NOTE:</b> Agree.</p> <p>– <b>CHANGED:</b> No change.</p>	C
181	K8		- Some VSS constructions may require security properties on channels between parties. For example, [Fel87] (a scheme referenced in the document) assumes a private channel from each player to every other player.	Channels: <a href="#">H6</a> , <a href="#">K8</a>	<p>– <b>NOTE:</b> We mention that schemes can have different properties based on the system model, including the communication model.</p> <p>– <b>CHANGED:</b> No change.</p>	—

#	Ref	Old location	K: Comments by Thalia May Laing (HP Inc.)	Related	Reply Notes	Rev
182	K9		- Proactive schemes may require some method of recognising whether a party has been corrupted and a way to 'reset' a corrupted party, so it is no longer corrupted,	J16, K9	– <b>NOTE:</b> Both “reactive” and “proactive” settings are relevant, with respect to “recognizing” corruptions and initiating rejuvenation of components. – <b>CHANGED:</b> No change.	—
183	K10		More generally, it may also be useful to highlight that the dealer must securely delegate shares of the secret to the players in all settings.	E65, I14, K10	– <b>NOTE:</b> The ability to handle untrusted dealers may also be relevant. – <b>CHANGED:</b> No change, but see items E65 and I14.	—
184	K11	247-248	(3) It may improve the reader’s understanding if the notation is consistent throughout. Specifically, k has two meanings throughout the document, which can lead to confusion. In Section 1, an ‘f out of n’ scheme is referred to (lines 247 and 248), meaning that the scheme is resilient to up to f out of n parties being compromised. In Sections 2.1 and 2.2, it is a ‘k out of n’ scheme (used in the sense that k-1 parties together do not know anything about the secret) and then in Section 2.5 the ‘f’ notation is used, while Section 3.1 is back to k. At this early stage in the document, is it true that f = k-1? Would it make sense to use one symbol (such as just k, and replace f with k-1) until it is necessary to split them?	A2, E45, E60, F2, K11, K12	– <b>NOTE:</b> We agree that a clarification is useful. – <b>CHANGED:</b> See reply to A2.	R3, R13, R17, R18, R49, R74, R72
185	K12	995-996	In Section 5.1 (lines 982-983), f is again used to denote the number of bad components and k is the minimum required number of good components. This use of k is conflicting with the notation used earlier in the document and could lead to confusion. In lines 995-996, it is said that in an n out of n signature scheme, f=n-1 while k=1. If f is the number of parties that can be compromised then it is clear that f=n-1 and, using your more recent definition of k, it is clear that k=1. But if k is used according to the original definition in the document, then k=n in this example as n players are required to compute a signature. Harmonising the notation throughout will help make the text clearer.	A2, E45, E60, F2, K11, K12	– <b>NOTE:</b> This set of paragraphs shows how <i>f</i> and <i>k</i> depend on the security property at stake. It purposely shows how the same scheme can have different values <i>k</i> for different properties. – <b>CHANGED:</b> Moved the explanation of possible/omitted indices (C, A, I) to earlier in section 5.1 (now within section 5.1.1) to make it more clear upfront. Also added indices in more uses of <i>f</i> and <i>k</i> .	R73, R74

#	Ref	Old location	L: Comments by Karim Eldefrawy (SRI International)	Related	Reply Notes	Rev
186	L1		The NISTIR 8214 document has done a great job covering most aspects involved in threshold cryptographic schemes, e.g., adversary and system model, dynamic groups, connectivity between devices, proactive security guarantees.	D1, E1, F1, G1, H2, H8, I1, J1, K1, L1, M2	<p>– <b>NOTE:</b> Thank you for the encouragement.</p> <p>– <b>CHANGED:</b> No change.</p>	—
187	L2		An additional issue to consider is that of <b>mixed adversaries</b> . Most existing proactive secret sharing (PSS) schemes only guarantee secrecy in the presence of an honest majority with at most $n/2 - 1$ total corruptions during such a refresh period; an adversary that corrupts a single additional party beyond the $n/2 - 1$ threshold, even if only passively and only temporarily, obtains the secret.	H6, I7, L2, L3	<p>– <b>NOTE:</b> The aspect of <a href="#">graceful degradation</a> mentioned in Sec. 4.3.4 can also be considered for the case where the compromise threshold is exceeded.</p> <p>– <b>CHANGED:</b> See reply to item L3.</p>	R66
188	L3		Recent work [DELOY16] developed a PSS scheme secure in the presence of a dishonest majority. The PSS scheme is robust and secure against $t < n-2$ passive adversaries when there are no active corruptions, and secure but non-robust (but with identifiable aborts) against $t < n/2-1$ active adversaries when there are no additional passive corruptions. The scheme is also secure (with identifiable aborts) against mixed adversaries controlling a combination of passively and actively corrupted parties such that if there are $k$ active corruptions there are less than $n-k-1$ total corruptions. The PSS scheme was then utilized to demonstrate feasibility of Proactive Secure Multiparty Computation (PMPC) with a dishonest majority with similar security guarantees. [DELOY16] Dolev S., Eldefrawy K., Lampkins J., Ostrovsky R., Yung M. (2016) Proactive Secret Sharing with a Dishonest Majority. In: Zikas V., De Prisco R. (eds) Security and Cryptography for Networks. SCN 2016. Lecture Notes in Computer Science, vol 9841. <a href="https://link.springer.com/chapter/10.1007/978-3-319-44618-9_28">https://link.springer.com/chapter/10.1007/978-3-319-44618-9_28</a>	H6, I7, L2, L3	<p>– <b>NOTE:</b> It is pertinent to compare how a protocol behaves under different types of attackers (passive vs. active), and what kind of (graceful?) degradation may exist when a threshold of compromise is surpassed.</p> <p>– <b>CHANGED:</b> Added statement about dual thresholds in the end of Sec. 4.3.4. See also the reply to item L4.</p>	R66

#	Ref	Old location	L: Comments by Karim Eldefrawy (SRI International)	Related	Reply Notes	Rev
189	L4		Ongoing work is improving the communication and computation complexity of such PMPC scheme and by the time of the 2019 NIST workshop on threshold cryptography, more efficient PMPC schemes for dishonest majority should be publicly available. We plan to submit a workshop talk proposal overviewing the above issue, and these PSS and MPC protocols and modifications thereof that improve their communication and computation complexity and render them much more practical. [EOPY18] Eldefrawy K., Ostrovsky R., Park S., Yung M. (2018) Proactive Secure Multiparty Computation with a Dishonest Majority. In: Catalano D., De Prisco R. (eds) Security and Cryptography for Networks. SCN 2018. Lecture Notes in Computer Science, vol 11035. <a href="https://link.springer.com/chapter/10.1007/978-3-319-98113-0_11">https://link.springer.com/chapter/10.1007/978-3-319-98113-0_11</a>		<p>– <b>NOTE:</b> We welcome submissions to NTCW'19.</p> <p>– <b>CHANGED:</b> No change.</p>	—

#	Ref	Old location	M: Comments by John Wallrabenstein (Analog)	Related	Reply Notes	Rev
190	M1		A substantial barrier to the productization and adoption of novel cryptographic constructions is the requirement from customers to comply with existing NIST standards, particularly FIPS 140-2. While the list of standardized algorithms covers the core components of cryptographic systems, it remains a small subset of the cryptographic primitives discussed in peer-reviewed literature. This limitation beneficially ensures that algorithms are only adopted after careful cryptanalysis. However, many well-studied cryptographic primitives (e.g., threshold schemes, zero knowledge proofs, etc.) remain unstandardized and are therefore onerous to integrate into products targeting heavily regulated markets despite their ability to address known adversarial attack strategies.	<a href="#">B1</a> , <a href="#">G2</a> , <a href="#">I2</a> , <a href="#">M1</a>	<p>– <b>NOTE:</b> the present document intends to promote standardization of threshold schemes</p> <p>– <b>CHANGED:</b> No change</p>	—
191	M2		Analog Devices, Inc. supports the goal of Draft NISTIR 8214 to standardize threshold cryptographic schemes, which provide resiliency against a wide variety of real-world adversarial attack strategies.	<a href="#">D1</a> , <a href="#">E1</a> , <a href="#">F1</a> , <a href="#">G1</a> , <a href="#">H2</a> , <a href="#">H8</a> , <a href="#">I1</a> , <a href="#">J1</a> , <a href="#">K1</a> , <a href="#">L1</a> , <a href="#">M2</a>	<p>– <b>NOTE:</b> Thank you for the encouragement.</p> <p>– <b>CHANGED:</b> No change.</p>	—
192	M3		* On the granularity of certification, we suggest certifying individual primitives (e.g., threshold ECIES, distributed key generation, etc.) rather than larger composed constructions. This provides implementers a high degree of freedom in tailoring threshold systems to their specific application, and reduces the burden on the standard itself.	<a href="#">G5</a> , <a href="#">I6</a> , <a href="#">J7</a> , <a href="#">M3</a> , <a href="#">M4</a>	<p>– <b>NOTE:</b> Composability is an important matter to consider; we intend to promote standardization of threshold schemes for diverse cryptographic primitives.</p> <p>– <b>CHANGED:</b> In Sec. 7.2, a new <a href="#">paragraph</a> highlights the caution needed with composability of modular components.</p>	<a href="#">R108</a>
193	M4		The existing FIPS 140-2 standard already places the burden of ensuring primitives are combined in a secure way onto the implementer, and we suggest the same approach for a threshold standard.	<a href="#">G5</a> , <a href="#">I6</a> , <a href="#">J7</a> , <a href="#">M3</a> , <a href="#">M4</a>	<p>– <b>NOTE:</b> Somewhat unclear what is the suggested approach — let implementers combine primitives into a threshold scheme, or let implementers combine threshold schemes into higher-level applications? Composability is an important matter to consider.</p> <p>– <b>CHANGED:</b> See reply to <a href="#">M3</a>.</p>	<a href="#">R108</a>
194	M5		* It may be useful to first focus on standardizing threshold implementations of RSA, elliptic curve cryptography, and AES for the single device setting. The communication, consensus, and automated validation requirements for the multi-device setting appear to introduce far more obstacles to standardization than the single device setting, which removes many of these issues.		<p>– <b>NOTE:</b> We have identified both single-device and multi-party settings as part of the scope of promoting threshold schemes for cryptographic primitives. Different obstacles in different settings may be tackled in parallel.</p> <p>– <b>CHANGED:</b> No change.</p>	—

#	Ref	Old location	M: Comments by John Wallrabenstein (Analog)	Related	Reply Notes	Rev
195	M6		* A standard covering threshold versions of existing standardized cryptographic primitives in the single device setting also allows easier integration with existing NIST test harnesses for their corresponding non-threshold versions.	G3, M6	<p>– <b>NOTE:</b> Test and validation procedures may possibly come to depend on the characterizing features of threshold schemes</p> <p>– <b>CHANGED:</b> In section 6.2, added text mentioning that “the process towards standardization of threshold schemes may involve reconsidering the adequacy of the validation requirements and where necessary devise new of complementary requirements”.</p>	R89
196	M7	Lines 577-579	* Distributed key generation was briefly mentioned (lines 577-579, Pedersen '91) in the draft, although the context appears to consider only the multi-device setting. We suggest that in addition to the multi-device setting, distributed key generation for the single device setting (where all parties in the “distributed” key generation protocol reside on a single device) be standardized.	E59, M7	<p>– <b>NOTE:</b> Distributed key generation is relevant for both single-device and multiparty settings. See reply to E59.</p> <p>– <b>CHANGED:</b> No change.</p>	
197	M8		* Is there a tentative timeline and roadmap for arriving at a NIST standard(s) for threshold cryptographic schemes?		<p>– <b>NOTE:</b> This NISTIR was an initial step, to be followed by the NIST Threshold Cryptography Workshop 2019 (March 11–12). A tentative timeline and roadmap does not yet exist. We welcome feedback from stakeholders.</p> <p>– <b>CHANGED:</b> No change.</p>	C

#	Ref	Old location	N: Comments by authors (NIST) — editorial	Related	Reply Notes	Rev
198	N1	Abstract, line 50	Clarify upfront that threshold schemes are composed of multiple components that contribute to the intended outcome.		– <b>CHANGED:</b> Adjust sentence that mentions components for the first time (R1).	
199	N2	Line 1029	Add acknowledgments for public comments.		– <b>CHANGED:</b> Added acknowledgments to the contributors of public comments (R2).	
200	N3		Various editorial revisions: avoid contractions (apostrophes), correct typos, commas, define all acronyms, ...		– <b>CHANGED:</b> apostrophe contractions (R37,R39,R48,R90); commas (R46); missing verb (R87); citation tags as words (R78, R79); others (R85); “down time” (R86); RSA and AES (R4); CPU, SGX, ARM (R81); HSM (R80); RNG (R70).	
201	N4	Section 7.1	Reorganize the set of representative questions, considering their increased number.		– <b>CHANGED:</b> Organized the representative questions by topic; the indexing of questions is now made with a lower case letter (topic) and number (R92).	
202	N5	Line 1104, 1115	For better referencing, some portions of text can be promoted to numbered subsections or subsubsections.	E20	– <b>CHANGED:</b> Several subsections of Sections 3, 4 and 5 now have numbered subsubsections; Section 7 now has numbered subsections.	
203	N6	Line 1029	Blakley scheme: shares in 2-out-of- $n$ Blakley must be non-vertical; shares in $k$ -out-of- $n$ must be non-orthogonal to the $x_1$ axis.		– <b>CHANGED:</b> Made explicit the orthogonality requirement: R27, R28, R30, R21 Added sentence selecting $x_1$ as the coordinate for the secret (R22).	
204	N7	Line 356	For the $k$ -out-of- $n$ Blakley scheme with $k > 2$ , explain how to extract the secret from the intersection point $P$ .	N7, E53	– <b>CHANGED:</b> Added sentence selecting $x_1$ as the coordinate (of $P$ ) that defines the secret (R22).	
205	N8		Some relocations across paragraphs.		– <b>CHANGED:</b> Old subsection “2.5 Terminology” is now Sec. 2.1, at the beginning of Section “Fundamentals” (R16); note on ISO/IEC secret-sharing is moved up a few paragraphs, to the end of new Sec. 2.2 (R26); In Sec. 2.2, note on $n$ vs. $Q$ is moved up one paragraph R19. Old 2nd paragraph of section 8 (conclusions) moved to (and then adjusted in) 1st paragraph of new Sec. 7.3 (R110).	

#	Ref	Old location	N: Comments by authors (NIST) — editorial	Related	Reply Notes	Rev
206	N9		Some text adjustments.		<p>– <b>CHANGED:</b> cite more recent attack (Foreshadow) (R9); “key” → “secret” (R25); cite previous descriptions of threshold RSA and mention “dealer” (R36); “universal” → “global” (R42); in section 4, old line 736, clarify [low thresholds] “be dealt with at a different application layer” (R55); replace “meta-questions” by “questions about ...” (R106); replace “centralized authority” by “same entity” (R63); various other text improvements (R6, R8, R10, R14, R35, R40, R47, R50, R75, R76).</p>	
207	N10	After line 760	Connect the importance of security models / pitfalls of decoupling security properties to the case of threshold schemes.		<p>– <b>CHANGED:</b> Added corresponding paragraph (R56).</p>	