

Withdrawn Draft

Warning Notice

The attached draft document has been withdrawn, and is provided solely for historical purposes. It has been superseded by the document identified below.

Withdrawal Date August 17, 2022

Original Release Date October 18, 2021

Superseding Document

Stage Final

Series/Number NIST SP 800-108r1

Title Recommendation for Key Derivation Using Pseudorandom Functions

Publication Date August 2022

DOI <https://doi.org/10.6028/NIST.SP.800-108r1>

CSRC URL <https://csrc.nist.gov/publications/detail/sp/800-108/rev-1/final>

Additional Information

3 **Recommendation for Key Derivation**
4 **Using Pseudorandom Functions**

5
6
7 Lily Chen
8
9
10
11
12
13

14 This publication is available free of charge from:
15 <https://doi.org/10.6028/NIST.SP.800-108r1-draft>
16
17
18
19

20 **Draft NIST Special Publication 800-108**
21 **Revision 1**
22

23 **Recommendation for Key Derivation**
24 **Using Pseudorandom Functions**
25

26 Lily Chen
27 *Computer Security Division*
28 *Information Technology Laboratory*
29
30
31
32
33
34
35

36
37 This publication is available free of charge from:
38 <https://doi.org/10.6028/NIST.SP.800-108r1-draft>
39

40 October 2021
41



42
43
44
45 U.S. Department of Commerce
46 *Gina M. Raimondo, Secretary*
47

48 National Institute of Standards and Technology
49 *James K. Olthoff, Performing the Non-Exclusive Functions and Duties of the Under Secretary of Commerce*
50 *for Standards and Technology & Director, National Institute of Standards and Technology*

51

Authority

52 This publication has been developed by NIST in accordance with its statutory responsibilities under the
53 Federal Information Security Modernization Act (FISMA) of 2014, 44 U.S.C. § 3551 *et seq.*, Public Law
54 (P.L.) 113-283. NIST is responsible for developing information security standards and guidelines, including
55 minimum requirements for federal information systems, but such standards and guidelines shall not apply
56 to national security systems without the express approval of appropriate federal officials exercising policy
57 authority over such systems. This guideline is consistent with the requirements of the Office of Management
58 and Budget (OMB) Circular A-130.

59 Nothing in this publication should be taken to contradict the standards and guidelines made mandatory and
60 binding on federal agencies by the Secretary of Commerce under statutory authority. Nor should these
61 guidelines be interpreted as altering or superseding the existing authorities of the Secretary of Commerce,
62 Director of the OMB, or any other federal official. This publication may be used by nongovernmental
63 organizations on a voluntary basis and is not subject to copyright in the United States. Attribution would,
64 however, be appreciated by NIST.

65 National Institute of Standards and Technology Special Publication 800-108 Revision 1
66 Natl. Inst. Stand. Technol. Spec. Publ. 800-108 Rev. 1, 26 pages (October 2021)
67 CODEN: NSPUE2

68 This publication is available free of charge from:
69 <https://doi.org/10.6028/NIST.SP.800-108r1-draft>

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

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

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

83 **Public comment period: *October 18, 2021 through January 18, 2022***

84 National Institute of Standards and Technology
85 Attn: Computer Security Division, Information Technology Laboratory
86 100 Bureau Drive (Mail Stop 8930) Gaithersburg, MD 20899-8930
87 Email: sp800-108-comments@nist.gov

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

89

Reports on Computer Systems Technology

90 The Information Technology Laboratory (ITL) at the National Institute of Standards and
91 Technology (NIST) promotes the U.S. economy and public welfare by providing technical
92 leadership for the Nation’s measurement and standards infrastructure. ITL develops tests, test
93 methods, reference data, proof of concept implementations, and technical analyses to advance the
94 development and productive use of information technology. ITL’s responsibilities include the
95 development of management, administrative, technical, and physical standards and guidelines for
96 the cost-effective security and privacy of other than national security-related information in federal
97 information systems. The Special Publication 800-series reports on ITL’s research, guidelines, and
98 outreach efforts in information system security, and its collaborative activities with industry,
99 government, and academic organizations.

100

Abstract

101 This Recommendation specifies techniques for the derivation of additional keying material from
102 a secret key—either established through a key establishment scheme or shared through some
103 other manner—using pseudorandom functions HMAC, CMAC, and KMAC.

104

Keywords

105 CMAC; HMAC; key derivation; KMAC; pseudorandom function.

106

107

Acknowledgments

108 The author, Lily Chen of the National Institute of Standards and Technology (NIST) would like
109 to thank her colleagues – Elaine Barker and Meltem Sönmez Turan of NIST and Rich Davis of
110 the National Security Agency – for their helpful discussions and valuable comments. The author
111 also gratefully appreciates the comments received during the development of the previous
112 version of this publication.

113

Call for Patent Claims

114 This public review includes a call for information on essential patent claims (claims whose use
115 would be required for compliance with the guidance or requirements in this Information
116 Technology Laboratory (ITL) draft publication). Such guidance and/or requirements may be
117 directly stated in this ITL Publication or by reference to another publication. This call also
118 includes disclosure, where known, of the existence of pending U.S. or foreign patent applications
119 relating to this ITL draft publication and of any relevant unexpired U.S. or foreign patents.

120 ITL may require from the patent holder, or a party authorized to make assurances on its behalf,
121 in written or electronic form, either:

122 a) assurance in the form of a general disclaimer to the effect that such party does not hold
123 and does not currently intend holding any essential patent claim(s); or

124 b) assurance that a license to such essential patent claim(s) will be made available to
125 applicants desiring to utilize the license for the purpose of complying with the guidance
126 or requirements in this ITL draft publication either:

127 i. under reasonable terms and conditions that are demonstrably free of any unfair
128 discrimination; or

129 ii. without compensation and under reasonable terms and conditions that are
130 demonstrably free of any unfair discrimination.

131 Such assurance shall indicate that the patent holder (or third party authorized to make assurances
132 on its behalf) will include in any documents transferring ownership of patents subject to the
133 assurance, provisions sufficient to ensure that the commitments in the assurance are binding on
134 the transferee, and that the transferee will similarly include appropriate provisions in the event of
135 future transfers with the goal of binding each successor-in-interest.

136 The assurance shall also indicate that it is intended to be binding on successors-in-interest
137 regardless of whether such provisions are included in the relevant transfer documents.

138 Such statements should be addressed to: sp800-108-comments@nist.gov

139

140 **Table of Contents**

141 **1 Introduction 1**

142 **2 Scope and Purpose..... 1**

143 **3 Definitions, Symbols, and Abbreviations 2**

144 3.1 Definitions 2

145 3.2 Symbols and Abbreviations 4

146 **4 Pseudorandom Function (PRF) 6**

147 **5 Key Derivation Function (KDF) 7**

148 5.1 KDF in Counter Mode 9

149 5.2 KDF in Feedback Mode 11

150 5.3 KDF in Double-Pipeline Mode..... 12

151 5.4 KDF Using KMAC 13

152 **6 Key Hierarchy 15**

153 **7 Security Considerations 16**

154 7.1 Cryptographic Strength 16

155 7.2 The Length of Key Derivation Key 16

156 7.3 Converting Keying Material to Cryptographic Keys..... 16

157 7.4 Input Data Encoding 17

158 7.5 Key Separation 17

159 7.6 Context Binding 18

160 **References 19**

161 **List of Appendices**

162

163 **Appendix A— Revisions 20**

164

165 **List of Figures**

166 Figure 1. KDF in Counter Mode 10

167 Figure 2. KDF in Feedback Mode 12

168 Figure 3. KDF in Double-pipeline Mode 13

169 Figure 4. KDF Using KMAC 14

170 Figure 5. Key Hierarchy..... 15

171

172 **1 Introduction**

173 When a party obtains a cryptographic key, additional keys will often be needed. There are
174 numerous methods for obtaining the keying material required by **approved** cryptographic
175 algorithms (see SP 800-133 Rev. 2 [1] for a discussion of the recommended techniques). The
176 requisite keying material is often obtained from the output of a key-derivation function that takes
177 a preexisting cryptographic key (and other data) as input. Key-derivation functions are used to
178 derive additional keys from a cryptographic key.

179 The key derivation functions specified in the original edition (2008) of NIST Special Publication
180 (SP) 800-108¹ used pseudorandom functions HMAC and CMAC. In Revision 1, KDF using
181 KMAC is added in Section 5.4.

182 **2 Scope and Purpose**

183 This Recommendation specifies several families of key derivation functions that use
184 pseudorandom functions. These key derivation functions can be used to derive additional keys
185 from an existing cryptographic key that was previously established through an automated key-
186 establishment scheme (e.g., as defined in SP 800-56A [2] and SP 800-56B [3]), previously
187 generated (e.g., using a pseudorandom bit generator as specified in SP 800-90A [4] or a previous
188 instance of key derivation as specified in this Recommendation), and/or previously shared in
189 some other way (e.g., by manual distribution).

190 Effectively, the key derivation functions specified in this Recommendation provide the key
191 expansion functionality described in [5], where key derivation is portrayed as a process that
192 potentially requires two separate steps: 1) randomness extraction (to obtain an initial key) and 2)
193 key expansion (to produce additional keys from that initial key and other data).

¹ Chen L (2008) Recommendation for Key Derivation Using Pseudorandom Functions. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-108.

194 **3 Definitions, Symbols, and Abbreviations**195 **3.1 Definitions**

approved	FIPS-approved or NIST-recommended. An algorithm or technique that is either 1) specified in a FIPS or NIST Recommendation, 2) adopted in a FIPS or NIST Recommendation, or 3) specified in a list of NIST-approved security functions.
cryptographic key	A bit string used as a secret parameter by a cryptographic algorithm. In this Recommendation, a cryptographic key is either a truly random bit string of a length specified by the cryptographic algorithm or a pseudorandom bit string of the specified length that is computationally indistinguishable from one selected uniformly at random from the set of all bit strings of that length.
entity	An individual (person), organization, device, or a combination thereof. In this Recommendation, an entity may be a functional unit that executes certain processes.
hash function	<p>A function that maps a bit string of arbitrary length to a fixed-length bit string. Approved hash functions satisfy the following properties:</p> <ol style="list-style-type: none"> 1. (Collision resistance) It is computationally infeasible to find any two distinct inputs that map to the same output. 2. (Preimage resistance) Given a randomly chosen target output, it is computationally infeasible to find any input that maps to that output. (This property is called the one-way property.) 3. (Second preimage resistance) Given one input value, it is computationally infeasible to find a second (distinct) input value that maps to the same output as the first value. <p>This Recommendation uses the strength of the preimage resistance of a hash function as a contributing factor when determining the security strength provided by a key-derivation method.</p>
key derivation	The process by which keying material is derived from 1) either a cryptographic key or a shared secret produced during a key-agreement scheme and 2) other data. This Recommendation specifies key derivation from a cryptographic key.
key-derivation function	A function that, with the input of a cryptographic key and other data, generates a bit string called the keying material, as defined in this Recommendation.

key-derivation key (KDK)	A key used as an input to a key derivation function to derive additional keying material.
key establishment	A procedure conducted by two or more participants, after which the resultant keying material is shared by all participants.
key hierarchy	A multiple-level tree structure such that each node represents a key, and each branch – pointing from one node to another – indicates a key derivation from one key to another key.
keying material	A bit string such that non-overlapping segments of the string (with the required lengths) can be used as cryptographic keys or other secret (pseudorandom) parameters.
message authentication code (MAC)	A family of secret-key cryptographic algorithms acting on input data of arbitrary length to produce an output value of a specified length (called the MAC of the input data), which can be employed to provide authentication of the origin of data and/or data-integrity protection. In this Recommendation, approved MAC algorithms are used to determine families of pseudorandom functions (indexed by the choice of key) that are employed during key derivation.
mode of iteration	A method for iterating the multiple invocations of a pseudorandom function in order to derive the keying material with a required length.
nonce	A time-varying value that has – at most – a negligible chance of repeating; for example, a random value that is generated anew for each use, a timestamp, a sequence number, or some combination of these.
pipeline	A term used to describe a series of sequential executions of a pseudorandom function.
pseudorandom function	An indexed family of (efficiently computable) functions, each defined for the same particular pair of input and output spaces. (For the purposes of this Recommendation, one may assume that both the index set and the output space are finite.) The indexed functions are pseudorandom in that if a function from the family is selected by choosing an index value uniformly at random, and one's knowledge of the selected function is limited to the output values corresponding to a feasible number of (adaptively) chosen input values, then the selected function is computationally indistinguishable from a function whose outputs were fixed uniformly at random.
security strength	A number characterizing the amount of work that is expected to suffice to “break” the security definition of a given cryptographic algorithm.

shall	The term used to indicate a requirement of a Federal Information Processing Standard (FIPS) or a requirement that needs to be fulfilled to claim conformance with this Recommendation. Note that shall may be coupled with not to become shall not .
should	The term used to indicate an important recommendation. Ignoring the recommendation could result in undesirable results. Note that should may be coupled with not to become should not .

196

197 **3.2 Symbols and Abbreviations**

$A(i)$	The output of the i^{th} iteration in the first pipeline in a double pipeline iteration mode.
$A B$	The concatenation of bit strings A and B .
CMAC	Cipher-based Message Authentication Code (as specified in NIST SP 800-38B [6]).
h	The length of the PRF output in bits.
HMAC	Keyed-hash Message Authentication Code (as specified in FIPS 198-1 [7]).
i	The counter incremented following each iteration of PRF evaluation; it is represented as a bit string of length r when it is used as an input to the PRF.
IV	A bit string that is used as an initial value in computing the first iteration of the PRF in feedback mode. It may be an empty string.
KDF	Key Derivation Function.
$K(i)$	The output of the i^{th} iteration of the PRF.
K_I	A key-derivation key. K_I is used as input to a key-derivation function (along with other data) in order to derive the output keying material K_O .
K_O	Output keying material that is derived from the key-derivation key K_I and other data that were used as input to a key-derivation function.
KDK	Key-derivation key.

KMAC	Keccak-based Message Authentication Code (as specified in SP 800-185 [8]).
L	An integer specifying the length of the derived keying material K_O in bits, which is represented as a bit string when it is an input to a key-derivation function.
MAC	Message Authentication Code.
n	The number of iterations of the PRF needed to generate L bits of keying material.
PRF	Pseudorandom Function.
$PRF(s, x)$	A pseudorandom function with seed s and input data x .
r	An integer that is less than or equal to 32 and whose value is the length of the binary representation of the counter i when i is an input in counter mode or (optionally) in feedback mode and double-pipeline iteration mode of each iteration of the PRF.
$ X $	The length of a bit string X in bits.
$[T]_2$	The length of an integer T when represented as a bit string.
w	The length of a key-derivation key in bits.
$\{X\}$	Used to indicate that data X is an optional input to the key-derivation function.
$\lceil X \rceil$	The smallest integer that is larger than or equal to X . The ceiling of X . For example, $\lceil 8.2 \rceil = 9$.
$X := Y$	X is defined to be equal to Y .
\emptyset	The empty bit string. That is, for any bit string A , $\emptyset \parallel A = A \parallel \emptyset = A$.
0x00	An all-zero octet.
\in	For an element s and a set S , $s \in S$, means s belongs to S .

199 **4 Pseudorandom Function (PRF)**

200 A pseudorandom function (PRF) is the basic building block in constructing a key derivation
201 function in this Recommendation. Generally, a PRF family $\{PRF(s, x) \mid s \in S\}$ consists of
202 polynomial time computable functions with an index (also called a seed) s and input x , such that
203 when s is randomly selected from S and not known to observers, $PRF(s, x)$ is computationally
204 indistinguishable from a random function defined on the same domain with output to the same
205 range as $PRF(s, x)$. For a formal definition of a pseudorandom function, refer to [9].

206 When a cryptographic key K_I is regarded as the seed, that is, $s = K_I$, the output of the
207 pseudorandom function can be used as keying material. In Section 5, several families of PRF-
208 based key derivation functions are defined without describing the internal structure of the PRF.
209 For key derivation, this Recommendation approves the use of the keyed-Hash Message
210 Authentication Code (HMAC) specified in [7], the Cipher-based Message Authentication Code
211 (CMAC) specified in [6], and the Keccak-based Message Authentication Code (KMAC)
212 specified in [8] as the pseudorandom function. For a given KDF using HMAC, CMAC, or
213 KMAC, the key K_I is assumed to be computationally indistinguishable from one that has been
214 selected uniformly at random from the set of all of the bit strings with length of $|K_I|$.

215 Note that [5] specifies key-derivation methods that can be employed only as components of key-
216 agreement schemes, as described in [2] and [3].

5 Key Derivation Function (KDF)

218 This section defines several families of key-derivation functions (KDF) that use PRFs. For the
219 purposes of this Recommendation, a KDF is a function that – given input consisting of a (secret)
220 key and other data – is used to generate (i.e., derive) keying material that can be employed by
221 cryptographic algorithms. In other words, the KDFs specified here provide a key-expansion
222 capability (as noted in Section 2).

223 The key that is input to a key-derivation function is called a key-derivation key (KDK). To
224 comply with this Recommendation, a KDK **shall** be a cryptographic key (see Section 3.1). The
225 KDK used as an input to one of the key derivation functions specified in this Recommendation
226 can, for example, be generated by an approved cryptographic random bit generator (e.g., by a
227 deterministic random bit generator of the type specified in [4] or output by an approved
228 automated key-establishment scheme (e.g., as defined in [2] and [3]). The KDK can be a portion
229 of the keying material derived from another KDK.

230 Note that the key-derivation methods employed as components of key-agreement schemes (as
231 described in [2], [3], and [5]) include two-step methods in which the first step consists of
232 extracting a KDK from a shared secret precursor. These extracted KDKs are not part of the
233 output of a key-agreement scheme; they are only used to derive output keying material during a
234 single execution of a scheme and then destroyed (along with all other sensitive, locally stored
235 data associated with that particular execution).

236 In keeping with the usual terminology, the output of a key-derivation function is called the
237 derived keying material and may subsequently be segmented into multiple keys. Any disjointed
238 segments of the derived keying material (with the required lengths) can be used as cryptographic
239 keys for the intended algorithms. However, in order to make sure that different parties will
240 obtain the same keys from the derived keying material or, in the case where a single party
241 derives the keying material, that re-derivation will generate the same keys (when required), the
242 cryptographic application employing a KDF must define the way to convert (i.e., parse) the
243 keying material into different keys. For example, when 256 bits of keying material are derived,
244 the application may specify that the first 128 bits will be used as a key for a message
245 authentication code and that the second 128 bits will be used as an encryption key for a given
246 encryption algorithm.

247 Depending on the intended length of the keying material to be derived, the KDF may require
248 multiple invocations of the PRF used in its construction. A method for iterating the multiple
249 invocations is called a mode of iteration. In this Recommendation, a counter mode, a feedback
250 mode, and a double pipeline mode are specified in Sections 5.1, 5.2, and 5.3, respectively.

251 In addition to these iteration modes, this Recommendation specifies a KDF using KMAC in
252 Section 5.4. KMAC can output keying material that has the required length without iteration.

253 To define key-derivation functions, the following notations are used. Some of the notations have
254 been defined in Section 3.2 and are repeated here for easy reference.

- 255 1) K_I – Key-derivation key; a key that is used as an input to a key derivation function (along
256 with other input data) to derive keying material. When HMAC is used as the PRF, K_I is
257 used as the HMAC key, and the other input data is used as the value of $text$, as defined in
258 [7]. When CMAC is used as the PRF, K_I is used as the block cipher key, and the other
259 input data is used as the message M , as defined in [6]. When KMAC is used as the PRF,
260 K_I is used as the KMAC key, and the other input data is used as the main input string X ,
261 as defined in [8].
- 262 2) K_O – Keying material that is output from a key-derivation function specified in this
263 Recommendation; a bit string of the required length that is derived using a key-derivation
264 key (and other data).
- 265 3) $Label$ – A string that identifies the purpose for the derived keying material, which is
266 encoded as a bit string. The encoding method for the $Label$ is defined in a larger context,
267 for example, in the protocol that uses a KDF.
- 268 4) $Context$ – A bit string containing the information related to the derived keying material. It
269 may include the identities of the parties who are deriving and/or using the derived keying
270 material and, optionally, a nonce known by the parties who derive the keys.
- 271 5) IV – A bit string that is used as an initial value in computing the first iteration in the
272 feedback mode. It can be either public or secret. It may be an empty string. The length for
273 an IV should be specified by the application or protocol using the key-derivation
274 function.
- 275 6) L – An integer specifying the requested length (in bits) of the derived keying material K_O .
276 L is represented as a bit string when it is an input to a key-derivation function. The length
277 of the bit string is specified by the encoding method for the input data.
- 278 7) h – An integer that indicates the length (in bits) of the output of the PRF.
- 279 8) n – An integer whose value is the number of iterations of the PRF needed to generate L
280 bits of keying material.
- 281 9) i – A counter; a bit string of length r that is an input to each iteration of a PRF in the
282 counter mode and (optionally) in the feedback and double-pipeline iteration modes.
- 283 10) r – An integer ($r \leq 32$) that indicates the length of the binary representation of the
284 counter i .
- 285 11) $\{X\}$ – Used to indicate that the data X is an optional input to the key-derivation function.

286 12) *0x00* – An all-zero octet; an optional data field that is used to indicate a separation of
287 different variable-length data fields.²

288 For a PRF with an output length of h bits, the key-derivation function iterates the PRF n times,
289 concatenating the outputs until L bits of keying material are derived; this requires $n = \lceil L/h \rceil$.
290 When using counter mode, n **shall not** be larger than $2^r - 1$, where $r \leq 32$ is the length of the
291 binary representations of the counter values. This ensures that the counter values are distinct,
292 which is necessary to prevent a PRF used in counter mode from generating the same output. For
293 feedback mode and double-pipeline iteration mode, a repeat in the counter value (if a counter is
294 used at all) will not be sufficient to cause the iterated PRF to repeat an output value.
295 Nevertheless, for compliance with this Recommendation, n **shall not** be larger than $2^{32} - 1$ when
296 using feedback mode or double-pipeline iteration mode; $L = (2^{32} - 1)h$ bits of keying material is
297 more than enough for most applications. Regardless of the mode, a particular implementation of
298 a KDF or an application that uses a KDF can impose a smaller bound on the maximum
299 value of n (the number of PRF iterations) than those imposed here.

300 For each of the iterations of the PRF, the key-derivation key K_I is used as the key, and the input
301 data consists of an iteration-dependent input data and a string of fixed input data. Depending on
302 the mode of iteration, the iteration-dependent input data could be a counter, the output of the
303 PRF from the previous iteration, a combination of both, or an output from the first pipeline
304 iteration (in the case of double-pipeline iteration mode). In the following key-derivation
305 functions, the fixed input data is a concatenation of a *Label*, a separation indicator *0x00*, the
306 *Context*, and $[L]_2$. One or more of these fixed input data fields may be omitted unless required
307 for certain purposes, as discussed in Section 7.5 and Section 7.6.

308 The length for each data field and their order **shall** be defined unambiguously. For example, the
309 length and the order may be defined as part of a KDF specification or by the protocol where the
310 KDF is used. In each of the following sections, a specific order for the feedback value, the
311 counter, the *Label*, the separation indicator *0x00*, the *Context*, and $[L]_2$ is used, assuming that
312 each of them is represented with a specific length. This Recommendation specifies several
313 families of KDFs. Alternative orders for the input data fields may be used for different KDFs.

314 5.1 KDF in Counter Mode

315 This section specifies a family of KDFs that uses the counter mode. In counter mode, the output
316 of the PRF is computed with a counter as the iteration-dependent input data. The mode is defined
317 as follows.

318 Parameters:

- 319 • h – The length of the output of the PRF in bits

² This indicator may be considered as a part of the encoding method for the input data and can be replaced by other indicators (e.g., an indicator to represent the length of the variable length field). If, for a specific KDF, only data fields with identical lengths are used, then the indicator may be omitted.

- r – The length of the binary representation of the counter i

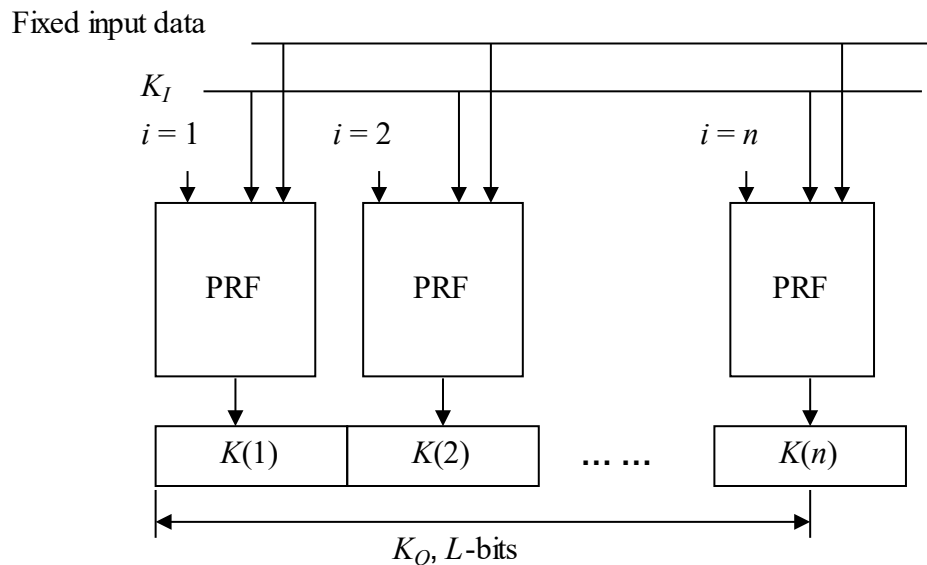
321 **Input:** K_I , $Label$, $Context$, and L .

322 **Process:**

- 323 1. $n := \lceil L/h \rceil$.
- 324 2. If $n > 2^r - 1$, then output an error indicator and stop (i.e., skip steps 3, 4, and 5).
- 325 3. $result := \emptyset$.
- 326 4. For $i = 1$ to n , do
 - 327 a. $K(i) := PRF(K_I, [i]_2 \parallel Label \parallel 0x00 \parallel Context \parallel [L]_2)$,
 - 328 b. $result = result \parallel K(i)$.
- 329 5. $K_O :=$ the leftmost L bits of $result$.

330 **Output:** K_O .

331 In each iteration of PRF evaluation in step 4 above, the fixed input data is the string $Label \parallel 0x00$
 332 $\parallel Context \parallel [L]_2$. The counter $[i]_2$ is the iteration-dependent input data and is represented as a bit
 333 string of r bits. The KDF in counter mode is illustrated in Figure 1.



334

335

Figure 1. KDF in Counter Mode

336

337 **5.2 KDF in Feedback Mode**

338 This section specifies a family of KDFs that uses the feedback mode. In the feedback mode, the
 339 output of the PRF is computed using the result of the previous iteration and, optionally, using a
 340 counter as an iteration-dependent input data. The mode is defined as follows. (Note that when L
 341 $\leq h$, $IV = \emptyset$, and the counter is used, the feedback mode will generate an output that is identical
 342 to the output of the counter mode specified in Section 5.1.)

343 **Parameters:**

- 344 • h – The length of the output of the PRF in bits
- 345 • r – The length of the binary representation of the counter i . r is specified only when a
 346 counter is used as an input

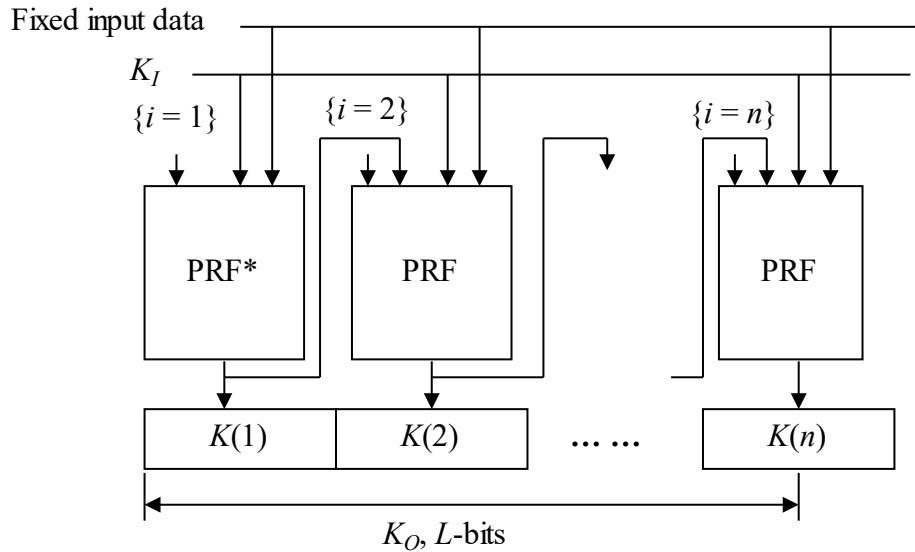
347 **Input:** K_I , $Label$, $Context$, IV , and L .

348 **Process:**

- 349 1. $n := \lceil L/h \rceil$.
- 350 2. If $n > 2^{32} - 1$, output an error indicator and stop (i.e., skip steps 3, 4, and 5).
- 351 3. $Result := \emptyset$ and $K(0) := IV$.
- 352 4. For $i = 1$ to n , do
 - 353 a. $K(i) := \text{PRF}(K_I, K(i-1) \{ \parallel [i]_2 \} \parallel Label \parallel 0x00 \parallel Context \parallel [L]_2)$.
 - 354 b. $result := result \parallel K(i)$.
- 355 5. $K_O :=$ the leftmost L bits of $result$.

356 **Output:** K_O .

357 In each iteration of PRF evaluation in step 4 above, the fixed input data is the string $Label \parallel 0x00$
 358 $\parallel Context \parallel [L]_2$. The iteration-dependent input data is $K(i-1) \{ \parallel [i]_2 \}$. The KDF in feedback mode
 359 is illustrated in Figure 2.



360

361

Figure 2. KDF in Feedback Mode

362 5.3 KDF in Double-Pipeline Mode

363 For a KDF in the counter mode or feedback mode, a PRF is iterated in a single pipeline. This
 364 section specifies a family of KDFs that iterates a PRF in two pipelines. In the first iteration
 365 pipeline, a sequence of secret values $A(i)$ is generated, each of which is used as an input to the
 366 respective PRF iteration in the second pipeline.

367 Parameters:

- 368 • h – The length of the output of the PRF in bits
- 369 • r – The length of the binary representation of the counter i . r is specified only when a
 370 counter is used as an input

371 **Input:** K_I , $Label$, $Context$, and L .

372 Process:

- 373 1. $n := \lceil L/h \rceil$.
- 374 2. If $n > 2^{32} - 1$, output an error indicator and stop (i.e., skip steps 3, 4, 5 and 6).
- 375 3. $Result := \emptyset$.
- 376 4. $A(0) := Label \parallel 0x00 \parallel Context \parallel [L]_2$.
- 377 5. For $i = 1$ to n , do
 - 378 a. $A(i) := PRF(K_I, A(i-1))$.

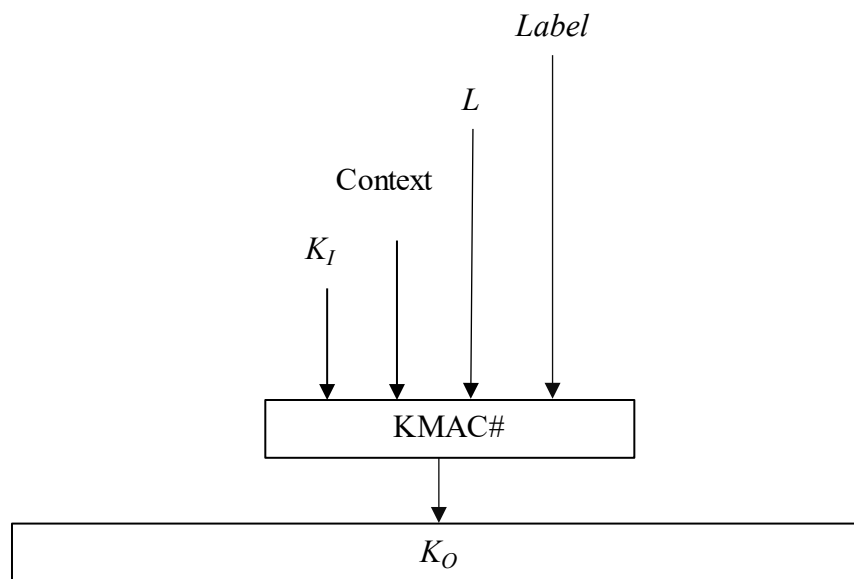
- 396 1. $K - K_I$, the key-derivation key.
- 397 2. $X - Context$, a bit string containing the information related to the derived keying material.
- 398 3. L – The desired output length of the derived keying material.
- 399 4. $S - Label$, an optional customization bit string; for example, $Label$ can be an encoding of
- 400 the characters “KDF” or “KDF4X” in 8-bit ASCII.

401 **Input:** K_I , $Context$, L , and $Label$.

402 **Process:**

- 403 1. If $L > 2^{32} - 1$, output an error indicator and stop (i.e., skip step 2).
- 404 2. $K_O = KMAC\#(K_I, Context, L, Label)$.

405 **Output:** K_O .



406

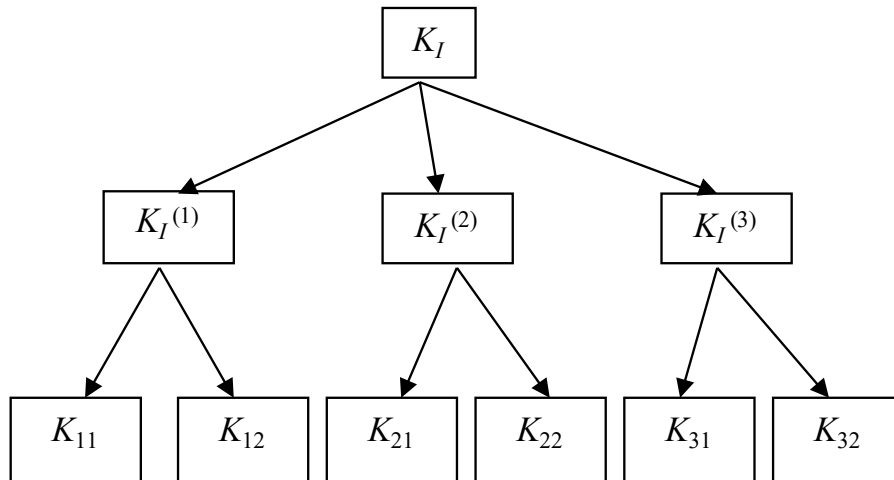
407

Figure 4. KDF Using KMAC

408

409 **6 Key Hierarchy**

410 The keying material derived from a given key-derivation key could subsequently be used as one
411 or more key-derivation keys to derive still more key-derivation keys. In this way, a key hierarchy
412 could be established. In a key hierarchy, a KDF is used with a higher-level “parent” key-
413 derivation key (and other appropriate input data) to derive a number of lower-level “child” keys.
414 Figure 5 presents a three-level key hierarchy as an example. In this example, the second level
415 keys $K_I^{(1)}$, $K_I^{(2)}$, and $K_I^{(3)}$ are derived from the top-level key K_I . Assuming that $K_I^{(1)}$, $K_I^{(2)}$, and
416 $K_I^{(3)}$ are used as key-derivation keys, further keys are derived from them as the bottom level keys
417 in the key hierarchy.



418

419

Figure 5. Key Hierarchy

420 **7 Security Considerations**

421 An improperly defined key-derivation function can make the derived keying material vulnerable
422 to attacks. This section will discuss some factors that affect the cryptographic strength of the
423 keying material derived by a KDF. However, some of the required security properties cannot be
424 achieved by the key-derivation function itself. For example, the overall security of the derived
425 keying material depends on the protocols that establish the key-derivation key. These external
426 conditions are out of the scope of the security discussion in this Recommendation.

427 **7.1 Cryptographic Strength**

428 The security strength of a key-derivation function is measured by the amount of work required to
429 distinguish the output of the KDF from a truly uniformly distributed bit string of the same length,
430 under the assumption that the key-derivation key, K_I , is the only unknown input to the KDF. This
431 is certainly no greater than the work required to recover K_I and/or the remaining portions of the
432 derived keying material from a given segment of KDF output. Given a set of input data (other
433 than K_I) and the corresponding output data of sufficient bit length (e.g., no less than w bits), the
434 key K_I can be recovered in (at most) 2^w executions of the KDF, where w is the bit length of K_I
435 through an exhaustive search over all possible K_I values.

436 **7.2 The Length of Key Derivation Key**

437 For some KDFs, the length of the key-derivation key is defined by the PRF used for the key
438 derivation. For example, when using CMAC as a PRF, the key length is uniquely determined by
439 the underlying block cipher. In this case, an implementation **should** check whether the key-
440 derivation key length is consistent with the length required by the PRF.

441 However, some PRFs can accommodate different key lengths. If HMAC is used as the PRF, then
442 a KDF can use a key-derivation key of essentially any length. It is worth noting, however, that if
443 the chosen key is longer than one input block for the hash function underlying HMAC, that key
444 will be hashed, and the (much shorter) h -bit output will be used as the HMAC key instead. In this
445 case, given a pair consisting of the input data (other than the key) and a sufficient amount of
446 corresponding output of the KDF, the hashed key can likely be recovered in (at most) 2^h
447 computations of the KDF. Therefore, the security strength of an HMAC-based key-derivation
448 function may actually be decreased by increasing the length of the KDK beyond the length of an
449 input block of the underlying hash function.

450 **7.3 Converting Keying Material to Cryptographic Keys**

451 The length of the derived keying material L depends on the requirements of the cryptographic
452 algorithms that rely on the KDF output. The length of a given cryptographic key is determined
453 by the algorithm that will employ it (e.g., a block cipher or a message authentication code) and
454 the desired security strength. In the absence of limitations that may be imposed by relying
455 applications, any segment of the derived keying material that has the required length can be
456 specified for use as a key, subject to the following restriction: when multiple keys (or any other
457 types of secret parameters, such as secret initialization vectors) are obtained from the derived
458 keying material, they **shall** be selected from disjointed (i.e., non-overlapping) segments of the
459 KDF output. Therefore, the value of L **shall** be greater than or equal to the sum of the lengths of

460 the keys and other types of secret parameters that will be obtained from the derived keying
461 material.

462 Note: To comply with this Recommendation, the derived keying material **shall not** be used as a
463 key stream for a stream cipher³ (i.e., using the derived keying material to encrypt through an
464 exclusive-or operation with plaintext is not permitted).

465 **7.4 Input Data Encoding**

466 The input data of a key-derivation function consists of different data fields (e.g., a *Label*, the
467 *Context*, and the length of the output keying material). In Section 5, each of the data fields
468 representing certain information is encoded as a bit string. The encoding method **shall** define a
469 one-to-one mapping from the set of all possible input information for that data field to a set of
470 the corresponding bit strings. The different data fields **shall** be assembled in a specific order. The
471 encoding method (including the field order) may be defined in a larger context (e.g., by the
472 protocol that uses a key-derivation function). The encoding method **shall** be designed for
473 unambiguous conversion of the combined input information to a unique bit string.

474 Unambiguous encoding for input data is required to deter attacks on the KDF that depend on
475 manipulating the input data. For detailed discussions on each attack, see [10].

476 **7.5 Key Separation**

477 In this Recommendation, key separation is a security requirement for the cryptographic keys
478 derived from the same key-derivation key. The keys **shall** be separate in the sense that the
479 compromise of some keys will not degrade the security strength of any of the other keys. In the
480 families of KDFs specified in this Recommendation, key separation can be achieved through
481 different approaches for the following two situations.

- 482 1. When keying material for multiple cryptographic keys is obtained from the output of a
483 single execution of a key-derivation function, the segments of the keying material used
484 for the different keys need to be cryptographically separate. The compromise of some
485 keys must not degrade the security of any of the other keys that are obtained from the
486 output of the same execution of a KDF. That is, the compromise of some keys must not
487 make the task of distinguishing any of the other keys from random strings with the same
488 length easier than the task would be if none of the keys were compromised. In order to
489 satisfy this requirement when using the key-derivation functions specified in this
490 Recommendation, different keys **shall** be obtained from disjointed (i.e., non-overlapping)
491 segments of the derived keying material.
- 492 2. When keying material for multiple cryptographic keys is obtained from the output of
493 multiple executions of a particular key-derivation function using the same value for K_I ,
494 the keying materials output by different calls to the KDF need to be cryptographically

³ The security strength provided by using the key-derivation functions specified in this Recommendation to generate a key stream for stream ciphers has not been investigated.

495 separate. The compromise of the keying material output from one of the executions of the
496 KDF must not degrade the security of any of the keying material output from the other
497 executions of the KDF. That is, the compromise must not make the task of distinguishing
498 any of the other keying material from random strings of the same length easier than the
499 task would be if none of the keying material were compromised. In order to satisfy this
500 requirement when using the key-derivation functions specified in this Recommendation,
501 different input data strings (e.g., *Label* || *0x00* || *Context* || [*L*]₂) **shall** be used for
502 different executions. The different data strings can be obtained by including different data
503 related to the derived keying materials. Examples of different information include:

- 504 – *Label*, if the keying materials are derived for different purposes.
- 505 – Identities included in *Context* if the keying materials are derived for different sets
506 of entities.
- 507 – A nonce included in the *Context* if the nonce is communicated by means of the
508 relying protocol and, therefore, shared by each entity who derives the keying
509 material; or
- 510 – Session identifiers if the keying materials are derived for different sessions.

511 7.6 Context Binding

512 Derived keying material **should** be bound to all relying entities and other information to identify
513 the derived keying material. This is called *context binding*. In particular, the identity (or
514 *identifier*, as the term is defined in [2] and [3]) of each entity that will access (i.e., derive, hold,
515 use, and/or distribute) any segment of the keying material **should** be included in the *Context*
516 string input to the KDF, provided that this information is known by each entity who derives the
517 keying material. In addition to identities, other information related to the derived keying material
518 (e.g., session identifiers, sequence numbers) as well as a nonce may be included in the *Context*
519 string, assuming that the information can be communicated, for instance, by means of the relying
520 protocol.

521 Context binding may not necessarily increase the security strength of an application making use
522 of a derived key. However, the binding may provide a way to detect protocol errors by providing
523 assurance that all parties who (correctly) derive the keying material are aware of who will access
524 it and in which session it will be used. If those parties have different understandings, then they
525 will derive different keying material. When that keying material is used in a protocol, the
526 protocol will likely fail to complete its execution and, therefore, will indicate errors to the
527 participants.

528

529 **References**

- 530 [1] Barker EB, Roginsky AL, Davis R (2020) Recommendation for Cryptographic Key
531 Generation. (National Institute of Standards and Technology, Gaithersburg, MD), NIST
532 Special Publication (SP) 800-133, Rev. 2. <https://doi.org/10.6028/NIST.SP.800-133r2>
- 533 [2] Barker EB, Chen L, Roginsky AL, Vassilev A, Davis R (2018) Recommendation for
534 Pair-Wise Key-Establishment Schemes Using Discrete Logarithm Cryptography.
535 (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special
536 Publication (SP) 800-56A, Rev. 3. <https://doi.org/10.6028/NIST.SP.800-56Ar3>
- 537 [3] Barker EB, Chen L, Roginsky AL, Vassilev A, Davis R, Simon S (2019)
538 Recommendation for Pair-Wise Key-Establishment Using Integer Factorization
539 Cryptography. (National Institute of Standards and Technology, Gaithersburg, MD),
540 NIST Special Publication (SP) 800-56B, Rev. 2. <https://doi.org/10.6028/NIST.SP.800-56Br2>
541
- 542 [4] Barker EB, Kelsey JM (2015) Recommendation for Random Number Generation Using
543 Deterministic Random Bit Generators. (National Institute of Standards and Technology,
544 Gaithersburg, MD), NIST Special Publication (SP) 800-90A, Rev. 1.
545 <https://doi.org/10.6028/NIST.SP.800-90Ar1>
- 546 [5] Barker EB, Chen L, Davis R (2020) Recommendation for Key-Derivation Methods in
547 Key-Establishment Schemes. (National Institute of Standards and Technology,
548 Gaithersburg, MD), NIST Special Publication (SP) 800-56C, Rev. 2.
549 <https://doi.org/10.6028/NIST.SP.800-56Cr2>
- 550 [6] Dworkin MJ (2005) Recommendation for Block Cipher Modes of Operation: the CMAC
551 Mode for Authentication. (National Institute of Standards and Technology, Gaithersburg,
552 MD), NIST Special Publication (SP) 800-38B, Includes updates as of October 6, 2016.
553 <https://doi.org/10.6028/NIST.SP.800-38B>
- 554 [7] National Institute of Standards and Technology (2008) The Keyed-Hash Message
555 Authentication Code (HMAC). (U.S. Department of Commerce, Washington, DC),
556 Federal Information Processing Standards Publication (FIPS) 198-1.
557 <https://doi.org/10.6028/NIST.FIPS.198-1>
- 558 [8] Kelsey JM, Chang S-jH, Perlner RA (2016) SHA-3 Derived Functions: cSHAKE,
559 KMAC, TupleHash, and ParallelHash. (National Institute of Standards and Technology,
560 Gaithersburg, MD), NIST Special Publication (SP) 800-185.
561 <https://doi.org/10.6028/NIST.SP.800-185>
- 562 [9] Goldreich O, Goldwasser S, Micali S (1986) How to construct pseudorandom functions,
563 *Journal of the ACM*. Vol. 33, No. 4, pp. 210-217. <https://doi.org/10.1145/6490.6503>
- 564 [10] Adams C, Kramer G, Mister S, Zuccherato R (2004) On the Security of Key Derivation
565 Functions. *Information Security* (Springer Verlag), LNCS 3225, pp. 134-145.
566 http://doi.org/10.1007/978-3-540-30144-8_12

567 Appendix A—Revisions

568 The original version of this document was published in November 2008. In October 2009, the
569 publication was updated with the following change:

570
571 In Section 5, page 12, at the end of the paragraph, “For each of the iterations of the PRF, the key
572 derivation key K_I is used as the key, and the input data consists of an iteration variable and a
573 string of fixed input data. Depending on the mode of iteration, the iteration variable could be a
574 counter, the output of the PRF from the previous iteration, a combination of both, or an output
575 from the first pipeline iteration in the case of double-pipeline iteration mode. In the following
576 key derivation functions, the fixed input data is a concatenation of a Label, a separation indicator
577 0x00, the Context, and $[L]_{2\dots}$ ”, the sentence “One or more of these fixed input data fields may
578 be omitted unless required for certain purposes, as discussed in Section 7.5 and Section 7.6.” was
579 added.