

# OCF Security Specification

VERSION 2.1.1 | February 2020



**OPEN** CONNECTIVITY  
FOUNDATION™

CONTACT [admin@openconnectivity.org](mailto:admin@openconnectivity.org)

Copyright Open Connectivity Foundation, Inc. © 2020.  
All Rights Reserved.

## 1 **LEGAL DISCLAIMER**

2 NOTHING CONTAINED IN THIS DOCUMENT SHALL BE DEEMED AS GRANTING YOU ANY KIND  
3 OF LICENSE IN ITS CONTENT, EITHER EXPRESSLY OR IMPLIEDLY, OR TO ANY  
4 INTELLECTUAL PROPERTY OWNED OR CONTROLLED BY ANY OF THE AUTHORS OR  
5 DEVELOPERS OF THIS DOCUMENT. THE INFORMATION CONTAINED HEREIN IS PROVIDED  
6 ON AN "AS IS" BASIS, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW,  
7 THE AUTHORS AND DEVELOPERS OF THIS SPECIFICATION HEREBY DISCLAIM ALL OTHER  
8 WARRANTIES AND CONDITIONS, EITHER EXPRESS OR IMPLIED, STATUTORY OR AT  
9 COMMON LAW, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OF  
10 MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. OPEN INTERCONNECT  
11 CONSORTIUM, INC. FURTHER DISCLAIMS ANY AND ALL WARRANTIES OF NON-  
12 INFRINGEMENT, ACCURACY OR LACK OF VIRUSES.

13 The OCF logo is a trademark of Open Connectivity Foundation, Inc. in the United States or other  
14 countries. \*Other names and brands may be claimed as the property of others.

15 Copyright © 2017-2020 Open Connectivity Foundation, Inc. All rights reserved.

16 Copying or other form of reproduction and/or distribution of these works are strictly prohibited

17 **CONTENTS**

18 1 Scope ..... 1

19 2 Normative References ..... 1

20 3 Terms, definitions, and abbreviated terms ..... 3

21 3.1 Terms and definitions..... 3

22 3.2 Abbreviated terms ..... 6

23 4 Document Conventions and Organization ..... 9

24 4.1 Conventions..... 9

25 4.2 Notation ..... 10

26 4.3 Data types ..... 11

27 4.4 Document structure..... 11

28 5 Security Overview..... 12

29 5.1 Preamble ..... 12

30 5.2 Access Control..... 14

31 5.2.1 ACL Architecture ..... 15

32 5.2.2 Access Control Scoping Levels..... 17

33 5.3 Onboarding Overview ..... 19

34 5.3.1 Onboarding General ..... 19

35 5.3.2 Onboarding Steps..... 21

36 5.3.3 Establishing a Device Owner ..... 22

37 5.3.4 Provisioning for Normal Operation ..... 23

38 5.3.5 Device Provisioning for OCF Cloud and Device Registration Overview –  
39 moved to OCF Cloud Security document ..... 23

40 5.3.6 OCF Compliance Management System..... 23

41 5.4 Provisioning..... 23

42 5.4.1 Provisioning General ..... 23

43 5.4.2 Provisioning other services..... 24

44 5.4.3 Provisioning Credentials for Normal Operation ..... 24

45 5.4.4 Role Assignment and Provisioning for Normal Operation ..... 24

46 5.4.5 ACL provisioning ..... 25

47 5.5 Secure Resource Manager (SRM)..... 25

48 5.6 Credential Overview..... 25

49 6 Security for the Discovery Process ..... 27

50 6.1 Preamble ..... 27

51 6.2 Security Considerations for Discovery..... 27

52 7 Security Provisioning ..... 29

53 7.1 Device Identity ..... 29

54 7.1.1 General Device Identity ..... 29

55 7.1.2 Device Identity for Devices with UAID [Deprecated]..... 29

56 7.2 Device Ownership..... 29

57 7.3 Device Ownership Transfer Methods..... 30

58 7.3.1 OTM implementation requirements ..... 30

59 7.3.2 SharedKey Credential Calculation ..... 31

60	7.3.3	Certificate Credential Generation.....	32
61	7.3.4	Just-Works OTM.....	32
62	7.3.5	Random PIN based OTM.....	33
63	7.3.6	Manufacturer Certificate Based OTM.....	36
64	7.3.7	Vendor Specific OTMs.....	38
65	7.3.8	Establishing Owner Credentials.....	39
66	7.3.9	Security considerations regarding selecting an Ownership Transfer Method	
67		- Moved to OCF Onboarding Tool document.....	42
68	7.3.10	Security Profile Assignment.....	42
69	7.4	Provisioning.....	43
70	7.4.1	Provisioning Flows.....	43
71	7.5	Device Provisioning for OCF Cloud – moved to OCF Cloud Security document.....	45
72	8	Device Onboarding State Definitions.....	45
73	8.1	Device Onboarding General.....	45
74	8.2	Device Onboarding-Reset State Definition.....	46
75	8.3	Device Ready-for-OTM State Definition.....	47
76	8.4	Device Ready-for-Provisioning State Definition.....	47
77	8.5	Device Ready-for-Normal-Operation State Definition.....	47
78	8.6	Device Soft Reset State Definition.....	48
79	9	Security Credential Management.....	49
80	9.1	Preamble.....	49
81	9.2	Credential Lifecycle.....	49
82	9.2.1	Credential Lifecycle General.....	49
83	9.2.2	Creation.....	49
84	9.2.3	Deletion.....	49
85	9.2.4	Refresh.....	49
86	9.2.5	Revocation.....	49
87	9.3	Credential Types.....	50
88	9.3.1	Preamble.....	50
89	9.3.2	Pair-wise Symmetric Key Credentials.....	50
90	9.3.3	Group Symmetric Key Credentials.....	50
91	9.3.4	Asymmetric Authentication Key Credentials.....	51
92	9.3.5	Asymmetric Key Encryption Key Credentials.....	51
93	9.3.6	Certificate Credentials.....	51
94	9.3.7	Password Credentials.....	52
95	9.4	Certificate Based Key Management.....	52
96	9.4.1	Overview.....	52
97	9.4.2	X.509 Digital Certificate Profiles.....	52
98	9.4.3	Certificate Revocation List (CRL) Profile [Deprecated].....	61
99	9.4.4	Resource Model.....	61
100	9.4.5	Certificate Provisioning.....	62
101	9.4.6	CRL Provisioning [Deprecated].....	62
102	10	Device Authentication.....	64
103	10.1	Device Authentication General.....	64

104	10.2	Device Authentication with Symmetric Key Credentials .....	64
105	10.3	Device Authentication with Raw Asymmetric Key Credentials.....	64
106	10.4	Device Authentication with Certificates .....	64
107	10.4.1	Device Authentication with Certificates General.....	64
108	10.4.2	Role Assertion with Certificates .....	65
109	10.4.3	OCF PKI Roots .....	66
110	10.4.4	PKI Trust Store.....	66
111	10.4.5	Path Validation and extension processing.....	67
112	10.5	Device Authentication with OCF Cloud – moved to OCF Cloud Security	
113		document.....	68
114	11	Message Integrity and Confidentiality .....	69
115	11.1	Preamble .....	69
116	11.2	Session Protection with DTLS.....	69
117	11.2.1	DTLS Protection General.....	69
118	11.2.2	Unicast Session Semantics.....	69
119	11.2.3	Cloud Session Semantics – moved to OCF Cloud Security document .....	69
120	11.3	Cipher Suites .....	69
121	11.3.1	Cipher Suites General .....	69
122	11.3.2	Cipher Suites for Device Ownership Transfer .....	69
123	11.3.3	Cipher Suites for Symmetric Keys.....	70
124	11.3.4	Cipher Suites for Asymmetric Credentials.....	71
125	11.3.5	Cipher suites for OCF Cloud Credentials – moved to OCF Cloud Security	
126		document .....	71
127	12	Access Control .....	72
128	12.1	ACL Generation and Management .....	72
129	12.2	ACL Evaluation and Enforcement.....	72
130	12.2.1	ACL Evaluation and Enforcement General.....	72
131	12.2.2	Host Reference Matching .....	72
132	12.2.3	Resource Wildcard Matching .....	72
133	12.2.4	Multiple Criteria Matching .....	73
134	12.2.5	Subject Matching using Wildcards .....	73
135	12.2.6	Subject Matching using Roles.....	73
136	12.2.7	ACL Evaluation.....	74
137	13	Security Resources .....	76
138	13.1	Security Resources General .....	76
139	13.2	Device Owner Transfer Resource .....	77
140	13.2.1	Device Owner Transfer Resource General.....	77
141	13.2.2	OCF defined OTMs.....	80
142	13.3	Credential Resource .....	80
143	13.3.1	Credential Resource General.....	80
144	13.3.2	Properties of the Credential Resource .....	85
145	13.3.3	Key Formatting .....	87
146	13.3.4	Credential Refresh Method Details [Deprecated] .....	87
147	13.4	Certificate Revocation List .....	87

148	13.4.1	CRL Resource Definition [Deprecated] .....	87
149	13.5	ACL Resources .....	87
150	13.5.1	ACL Resources General .....	87
151	13.5.2	OCF Access Control List (ACL) BNF defines ACL structures. ....	88
152	13.5.3	ACL Resource .....	89
153	13.6	Access Manager ACL Resource [Deprecated] .....	94
154	13.7	Signed ACL Resource [Deprecated] .....	94
155	13.8	Provisioning Status Resource .....	94
156	13.9	Certificate Signing Request Resource .....	99
157	13.10	Roles Resource .....	100
158	13.11	Account Resource – moved to OCF Cloud Security document.....	101
159	13.12	Account Session Resource – moved to OCF Cloud Security document .....	101
160	13.13	Account Token Refresh Resource – moved to OCF Cloud Security document.....	101
161	13.14	Security Virtual Resources (SVRs) and Access Policy .....	101
162	13.15	SVRs, Discoverability and OCF Endpoints .....	101
163	13.16	Additional Privacy Consideration for Core Resources .....	102
164	13.17	Easy Setup Resource Device State.....	103
165	14	Security Hardening Guidelines/ Execution Environment Security .....	106
166	14.1	Preamble .....	106
167	14.2	Execution Environment Elements .....	106
168	14.2.1	Execution Environment Elements General .....	106
169	14.2.2	Secure Storage.....	106
170	14.2.3	Secure execution engine .....	109
171	14.2.4	Trusted input/output paths .....	109
172	14.2.5	Secure clock.....	109
173	14.2.6	Approved algorithms.....	109
174	14.2.7	Hardware tamper protection.....	110
175	14.3	Secure Boot.....	110
176	14.3.1	Concept of software module authentication.....	110
177	14.3.2	Secure Boot process .....	112
178	14.3.3	Robustness Requirements.....	112
179	14.4	Attestation .....	112
180	14.5	Software Update .....	112
181	14.5.1	Overview: .....	112
182	14.5.2	Recognition of Current Differences .....	113
183	14.5.3	Software Version Validation.....	114
184	14.5.4	Software Update.....	114
185	14.5.5	Recommended Usage.....	114
186	14.6	Non-OCF Endpoint interoperability.....	115
187	14.7	Security Levels .....	115
188	14.8	Security Profiles.....	116
189	14.8.1	Security Profiles General .....	116
190	14.8.2	Identification of Security Profiles (Normative) .....	116
191	14.8.3	Security Profiles .....	118

192	15	Device Type Specific Requirements.....	123
193	15.1	Bridging Security .....	123
194	15.1.1	Universal Requirements for Bridging to another Ecosystem .....	123
195	15.1.2	Additional Security Requirements specific to Bridged Protocols .....	124
196	Annex A	(informative) Access Control Examples.....	126
197	A.1	Example OCF ACL Resource .....	126
198	Annex B	(Informative) Execution Environment Security Profiles .....	127
199	Annex C	(normative) Resource Type definitions.....	128
200	C.1	List of Resource Type definitions .....	128
201	C.2	Access Control List-2 .....	128
202	C.2.1	Introduction .....	128
203	C.2.2	Well-known URI .....	128
204	C.2.3	Resource type .....	128
205	C.2.4	OpenAPI 2.0 definition.....	128
206	C.2.5	Property definition .....	136
207	C.2.6	CRUDN behaviour .....	137
208	C.3	Credential .....	137
209	C.3.1	Introduction .....	137
210	C.3.2	Well-known URI .....	137
211	C.3.3	Resource type .....	137
212	C.3.4	OpenAPI 2.0 definition.....	137
213	C.3.5	Property definition .....	147
214	C.3.6	CRUDN behaviour .....	147
215	C.4	Certificate Signing Request.....	148
216	C.4.1	Introduction .....	148
217	C.4.2	Well-known URI .....	148
218	C.4.3	Resource type .....	148
219	C.4.4	OpenAPI 2.0 definition.....	148
220	C.4.5	Property definition .....	149
221	C.4.6	CRUDN behaviour .....	150
222	C.5	Device Owner Transfer Method.....	150
223	C.5.1	Introduction .....	150
224	C.5.2	Well-known URI .....	150
225	C.5.3	Resource type .....	150
226	C.5.4	OpenAPI 2.0 definition.....	150
227	C.5.5	Property definition .....	154
228	C.5.6	CRUDN behaviour .....	156
229	C.6	Device Provisioning Status .....	156
230	C.6.1	Introduction .....	156
231	C.6.2	Well-known URI .....	156
232	C.6.3	Resource type .....	156
233	C.6.4	OpenAPI 2.0 definition.....	156
234	C.6.5	Property definition .....	160
235	C.6.6	CRUDN behaviour .....	164

236	C.7	Asserted Roles .....	165
237	C.7.1	Introduction .....	165
238	C.7.2	Well-known URI .....	165
239	C.7.3	Resource type .....	165
240	C.7.4	OpenAPI 2.0 definition.....	165
241	C.7.5	Property definition .....	174
242	C.7.6	CRUDN behaviour .....	174
243	C.8	Security Profile .....	174
244	C.8.1	Introduction .....	174
245	C.8.2	Well-known URI .....	174
246	C.8.3	Resource type .....	175
247	C.8.4	OpenAPI 2.0 definition.....	175
248	C.8.5	Property definition .....	177
249	C.8.6	CRUDN behaviour .....	177
250	Annex D (informative)	OID definitions .....	178
251	Annex E (informative)	Security considerations specific to Bridged Protocols .....	180
252	E.1	Security Considerations specific to the AllJoyn Protocol .....	180
253	E.2	Security Considerations specific to the Bluetooth LE Protocol.....	180
254	E.3	Security Considerations specific to the oneM2M Protocol .....	180
255	E.4	Security Considerations specific to the U+ Protocol .....	181
256	E.5	Security Considerations specific to the Z-Wave Protocol.....	181
257	E.6	Security Considerations specific to the Zigbee Protocol .....	182
258	E.7	Security Considerations specific to the the EnOcean Radio Protocol.....	183
259			



260	<b>FIGURES</b>	
261	Figure 1 – OCF Interaction.....	10
262	Figure 2 – OCF Layers .....	12
263	Figure 3 – OCF Security Enforcement Points .....	14
264	Figure 4 – Use case-1 showing simple ACL enforcement.....	16
265	Figure 5 – Use case 2: A policy for the requested Resource is missing.....	17
266	Figure 6 – Example Resource definition with opaque Properties .....	18
267	Figure 7 – Property Level Access Control .....	18
268	Figure 8 – Onboarding Overview.....	20
269	Figure 9 – OCF Onboarding Process .....	22
270	Figure 10 – OCF's SRM Architecture .....	25
271	Figure 11 – Discover New Device Sequence.....	30
272	Figure 12 – A Just Works OTM .....	32
273	Figure 13 – Random PIN-based OTM .....	34
274	Figure 14 – Manufacturer Certificate Based OTM Sequence .....	37
275	Figure 15 – Vendor-specific Owner Transfer Sequence.....	39
276	Figure 16 – Symmetric Owner Credential Provisioning Sequence .....	41
277	Figure 17 – Example of Client-directed provisioning.....	44
278	Figure 18 – Device state model.....	46
279	Figure 19 – Client-directed Certificate Transfer.....	62
280	Figure 20 – Asserting a role with a certificate role credential. ....	66
281	Figure 21 – OCF Security Resources.....	76
282	Figure 22 – "/oic/sec/cred" Resource and Properties.....	77
283	Figure 23 – "/oic/sec/acl2" Resource and Properties.....	77
284	Figure 24 – Example of Soft AP and Easy Setup Resource in different Device states .....	103
285	Figure 25 – Software Module Authentication .....	111
286	Figure 26 – Verification Software Module.....	111
287	Figure 27 – Software Module Authenticity .....	112
288	Figure 28 – State transitioning diagram for software download .....	113
289	Figure A-1 – Example "/oic/sec/acl2" Resource.....	126
290	Figure E-1 Security Considerations for BLE Bridge .....	180
291	Figure E-2 Security Considerations for Z-Wave Bridge.....	182
292	Figure E-3 Security Considerations for Zigbee Bridge .....	183
293	Figure E-4 Security Considerations for EnOcean Bridge .....	184
294		

295 **Tables**

296 Table 1 – Discover New Device Details..... 31

297 Table 2 – A Just Works OTM Details..... 33

298 Table 3 – Random PIN-based OTM Details..... 34

299 Table 4 – Manufacturer Certificate Based OTM Details ..... 38

300 Table 5 – Vendor-specific Owner Transfer Details ..... 39

301 Table 6 – Symmetric Owner Credential Assignment Details ..... 41

302 Table 7 – Steps describing Client -directed provisioning ..... 44

303 Table 8 – X.509 v1 fields for Root CA Certificates..... 53

304 Table 9 - X.509 v3 extensions for Root CA Certificates ..... 53

305 Table 10 - X.509 v1 fields for Intermediate CA Certificates ..... 54

306 Table 11 – X.509 v3 extensions for Intermediate CA Certificates ..... 54

307 Table 12 – X.509 v1 fields for End-Entity Certificates..... 55

308 Table 13 – X.509 v3 extensions for End-Entity Certificates ..... 55

309 Table 14 – ACE2 Wildcard Matching Strings Description..... 72

310 Table 15 – Definition of the "/oic/sec/doxm" Resource ..... 78

311 Table 16 – Properties of the "/oic/sec/doxm" Resource ..... 78

312 Table 17 – Properties of the "oic.sec.didtype" type ..... 79

313 Table 18 – Properties of the "oic.sec.doxmtype" type..... 80

314 Table 19 – Definition of the "/oic/sec/cred" Resource ..... 81

315 Table 20 – Properties of the "/oic/sec/cred" Resource..... 82

316 Table 21 – Properties of the "oic.sec.creds" Property..... 83

317 Table 22: Properties of the "oic.sec.credusagetype" Property ..... 84

318 Table 23 – Properties of the "oic.sec.pubdatatype" Property ..... 84

319 Table 24 – Properties of the "oic.sec.privdatatype" Property ..... 84

320 Table 25 – Properties of the "oic.sec.optdatatype" Property ..... 85

321 Table 26 – Definition of the "oic.sec.roletype" type. .... 85

322 Table 27 – 128-bit symmetric key ..... 87

323 Table 28 – 256-bit symmetric key ..... 87

324 Table 29 – BNF Definition of OCF ACL ..... 88

325 Table 30 – Value Definition of the "oic.sec.crudntype" Property ..... 90

326 Table 31 – Definition of the "oic/sec/acl2" Resource ..... 90

327 Table 32 – Properties of the "/oic/sec/acl2" Resource ..... 91

328 Table 33 – "oic.sec.ace2" data type definition. .... 92

329 Table 34 – "oic.sec.ace2.resource-ref" data type definition. .... 92

330 Table 35 – Value definition "oic.sec.conntype" Property..... 92

331 Table 36 – Definition of the "/oic/sec/pstat" Resource ..... 94

332 Table 37 – Properties of the "/oic/sec/pstat" Resource..... 95

333 Table 38 – Properties of the ".oic.sec.dostype" Property ..... 96

334	Table 39 – Definition of the "oic.sec.dpmttype" Property .....	98
335	Table 40 – Value Definition of the "oic.sec.dpmttype" Property (Low-Byte) .....	98
336	Table 41 – Value Definition of the "oic.sec.dpmttype" Property (High-Byte).....	98
337	Table 42 – Definition of the "oic.sec.pomtype" Property .....	98
338	Table 43 – Value Definition of the "oic.sec.pomtype" Property .....	99
339	Table 44 – Definition of the "/oic/sec/csr" Resource .....	99
340	Table 45 – Properties of the "oic.r.csr" Resource .....	99
341	Table 46 – Definition of the "/oic/sec/roles" Resource .....	101
342	Table 47 – Properties of the "/oic/sec/roles" Resource.....	101
343	Table 48 – Core Resource Properties Access Modes given various Device States .....	102
344	Table 49 – Examples of Sensitive Data.....	107
345	Table 50 – Description of the software update bits.....	113
346	Table 51 – Definition of the "/oic/sec/sp" Resource .....	117
347	Table 52 – Properties of the "/oic/sec/sp" Resource.....	117
348	Table 53 – Dependencies of VOD Behaviour on Bridge state, as clarification of	
349	accompanying text.....	124
350	Table B.1 – OCF Security Profile .....	127
351	Table C.1 – Alphabetized list of security resources .....	128
352	Table C-1 – The Property definitions of the Resource with type "rt" = "oic.r.acl2". .....	136
353	Table C-2 – The CRUDN operations of the Resource with type "rt" = "oic.r.acl2". .....	137
354	Table C-3 – The Property definitions of the Resource with type "rt" = "oic.r.cred". .....	147
355	Table C-4 – The CRUDN operations of the Resource with type "rt" = "oic.r.cred". .....	148
356	Table C-5 – The Property definitions of the Resource with type "rt" = "oic.r.csr".....	150
357	Table C-6 – The CRUDN operations of the Resource with type "rt" = "oic.r.csr". .....	150
358	Table C-7 – The Property definitions of the Resource with type "rt" = "oic.r.doxm". .....	154
359	Table C-8 – The CRUDN operations of the Resource with type "rt" = "oic.r.doxm". .....	156
360	Table C-9 – The Property definitions of the Resource with type "rt" = "oic.r.pstat". .....	160
361	Table C-10 – The CRUDN operations of the Resource with type "rt" = "oic.r.pstat". .....	165
362	Table C-11 – The Property definitions of the Resource with type "rt" = "oic.r.roles". .....	174
363	Table C-12 – The CRUDN operations of the Resource with type "rt" = "oic.r.roles". .....	174
364	Table C-13 – The Property definitions of the Resource with type "rt" = "oic.r.sp". .....	177
365	Table C-14 – The CRUDN operations of the Resource with type "rt" = "oic.r.sp". .....	177
366	Table E.1 GAP security mode .....	180
367	Table E.2 TLS 1.2 Cipher Suites used by U+ .....	181
368	Table E.3 Z-Wave Security Class.....	182
369	Table E.4 Zigbee 3.0 Security Levels to the Network, and Application Support layers .....	183
370	Table E.5 EnOcean Radio Protocol security levels.....	183
371		
372		

## 373 1 Scope

374 This document defines security objectives, philosophy, resources and mechanism that impacts  
375 OCF base layers of ISO/IEC 30118-1:2018. ISO/IEC 30118-1:2018 contains informative security  
376 content. The OCF Security Specification contains security normative content and may contain  
377 informative content related to the OCF base or other OCF documents.

## 378 2 Normative References

379 The following documents, in whole or in part, are normatively referenced in this document and are  
380 indispensable for its application. For dated references, only the edition cited applies. For undated  
381 references, the latest edition of the referenced document (including any amendments) applies.

382 ISO/IEC 30118-1:2018 Information technology -- Open Connectivity Foundation (OCF)  
383 Specification -- Part 1: Core specification  
384 <https://www.iso.org/standard/53238.html>  
385 Latest version available at:  
386 [https://openconnectivity.org/specs/OCF\\_Core\\_Specification.pdf](https://openconnectivity.org/specs/OCF_Core_Specification.pdf)

387 ISO/IEC 30118-3:2018 Information technology -- Open Connectivity Foundation (OCF)  
388 Specification -- Part 3: Bridging specification  
389 <https://www.iso.org/standard/74240.html>  
390 Latest version available at:  
391 [https://openconnectivity.org/specs/OCF\\_Bridging\\_Specification.pdf](https://openconnectivity.org/specs/OCF_Bridging_Specification.pdf)

392 OCF Wi-Fi Easy Setup, Information technology – Open Connectivity Foundation (OCF)  
393 Specification – Part 7: Wi-Fi Easy Setup specification  
394 Latest version available at:  
395 [https://openconnectivity.org/specs/OCF\\_Wi-Fi\\_Easy\\_Setup\\_Specification.pdf](https://openconnectivity.org/specs/OCF_Wi-Fi_Easy_Setup_Specification.pdf)

396 OCF Cloud Specification, Information technology – Open Connectivity Foundation (OCF)  
397 Specification – Part 8: Cloud Specification  
398 Latest version available at:  
399 [https://openconnectivity.org/specs/OCF\\_Cloud\\_Specification.pdf](https://openconnectivity.org/specs/OCF_Cloud_Specification.pdf)

400 JSON SCHEMA, draft version 4, <http://json-schema.org/latest/json-schema-core.html>.

401 IETF RFC 2315, *PKCS #7: Cryptographic Message Syntax Version 1.5*, March 1998,  
402 <https://tools.ietf.org/html/rfc2315>

403 IETF RFC 2898, *PKCS #5: Password-Based Cryptography Specification Version 2.0*, September  
404 2000, <https://tools.ietf.org/html/rfc2898>

405 IETF RFC 2986, *PKCS #10: Certification Request Syntax Specification Version 1.7*, November  
406 2000, <https://tools.ietf.org/html/rfc2986>

407 IETF RFC 4122, *A Universally Unique IDentifier (UUID) URN Namespace*, July 2005,  
408 <https://tools.ietf.org/html/rfc4122>

409 IETF RFC 4279, *Pre-Shared Key Ciphersuites for Transport Layer Security (TLS)*, December  
410 2005, <https://tools.ietf.org/html/rfc4279>

411 IETF RFC 4492, *Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security  
412 (TLS)*, May 2006, <https://tools.ietf.org/html/rfc4492>

413 IETF RFC 5246, *The Transport Layer Security (TLS) Protocol Version 1.2*, August 2008,  
414 <https://tools.ietf.org/html/rfc5246>

415 IETF RFC 5280, *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation*  
416 *List (CRL) Profile*, May 2008, <https://tools.ietf.org/html/rfc5280>

417 IETF RFC 5489, *ECDHE\_PSK Cipher Suites for Transport Layer Security (TLS)*, March 2009,  
418 <https://tools.ietf.org/html/rfc5489>

419 IETF RFC 5545, *Internet Calendaring and Scheduling Core Object Specification (iCalendar)*,  
420 September 2009, <https://tools.ietf.org/html/rfc5545>

421 IETF RFC 5755, *An Internet Attribute Certificate Profile for Authorization*, January 2010,  
422 <https://tools.ietf.org/html/rfc5755>

423 IETF RFC 6347, *Datagram Transport Layer Security Version 1.2*, January 2012,  
424 <https://tools.ietf.org/html/rfc6347>

425 IETF RFC 6655, *AES-CCM Cipher Suites for Transport Layer Security (TLS)*, July 2012,  
426 <https://tools.ietf.org/html/rfc6655>

427 IETF RFC 6749, *The OAuth 2.0 Authorization Framework*, October 2012,  
428 <https://tools.ietf.org/html/rfc6749>

429 IETF RFC 6750, *The OAuth 2.0 Authorization Framework: Bearer Token Usage*, October 2012,  
430 <https://tools.ietf.org/html/rfc6750>

431 IETF RFC 7228, *Terminology for Constrained-Node Networks*, May 2014,  
432 <https://tools.ietf.org/html/rfc7228>

433 IETF RFC 7250, *Using Raw Public Keys in Transport Layer Security (TLS) and Datagram*  
434 *Transport Layer Security (DTLS)*, June 2014, <https://tools.ietf.org/html/rfc7250>

435 IETF RFC 7251, *AES-CCM Elliptic Curve Cryptography (ECC) Cipher Suites for TLS*, June 2014,  
436 <https://tools.ietf.org/html/rfc7251>

437 IETF RFC 7515, *JSON Web Signature (JWS)*, May 2015, <https://tools.ietf.org/html/rfc7515>

438 IETF RFC 7519, *JSON Web Token (JWT)*, May 2015, <https://tools.ietf.org/html/rfc7519>

439 IETF RFC 8323, *CoAP (Constrained Application Protocol) over TCP, TLS, and WebSockets*,  
440 February 2018, <https://tools.ietf.org/html/rfc8323>

441 IETF RFC 8392, *CBOR Web Token (CWT)*, May 2018, <https://tools.ietf.org/html/rfc8392>

442 IETF RFC 8520, *Manufacturer Usage Description Specification*, Mar 2019,  
443 <https://tools.ietf.org/html/rfc8520>

444 oneM2M Release 3 Specifications, <http://www.onem2m.org/technical/published-drafts>

445 OpenAPI specification, aka *Swagger RESTful API Documentation Specification*, Version 2.0  
446 <https://github.com/OAI/OpenAPI-Specification/blob/master/versions/2.0.md>

447

448 **3 Terms, definitions, and abbreviated terms**

449 **3.1 Terms and definitions**

450 For the purposes of this document, the terms and definitions given in ISO/IEC 30118-1:2018 and  
451 the following apply.

452 ISO and IEC maintain terminological databases for use in standardization at the following  
453 addresses:

454 – ISO Online browsing platform: available at <https://www.iso.org/obp>

455 – IEC Electropedia: available at <http://www.electropedia.org/>

456 **3.1.1**

457 **Access Management Service (AMS)**

458 dynamically constructs ACL Resources in response to a Device Resource request.

459 Note 1 to entry: An AMS can evaluate access policies remotely and supply the result to a Server which allows or denies  
460 a pending access request. An AMS is authorised to provision ACL Resources.

461 **3.1.2**

462 **Access Token – moved to OCF Cloud Security document**

463 **3.1.3**

464 **Authorization Provider – moved to OCF Cloud Security document**

465 **3.1.4**

466 **Client**

467 Note 1 to entry: The details are defined in ISO/IEC 30118-1:2018.

468 **3.1.5**

469 **Credential Management Service (CMS)**

470 a name and Resource Type ("oic.sec.cms") given to a Device that is authorized to provision  
471 credential Resources.

472 **3.1.6**

473 **Device**

474 Note 1 to entry: The details are defined in ISO/IEC 30118-1:2018.

475 **3.1.7**

476 **Device Class**

477 Note 1 to entry: As defined in IETF RFC 7228. IETF RFC 7228 defines classes of constrained devices that distinguish  
478 when the OCF small footprint stack is used vs. a large footprint stack. Class 2 and below is for small footprint stacks.

479 **3.1.8**

480 **Device ID**

481 a stack instance identifier.

482 **3.1.9**

483 **Device Ownership Transfer Service (DOTS)**

484 a logical entity that establishes device ownership

485 **3.1.10**

486 **3.1.11 Device Registration – moved to OCF Cloud Security document**

487 **End-Entity**

488 any certificate holder which is not a Root or Intermediate Certificate Authority.

489 Note 1 to entry: Typically, a device certificate.

490 **3.1.12**

491 **Entity**

492 Note 1 to entry: The details are defined in ISO/IEC 30118-1:2018.

493 **3.1.13**  
494 **OCF Interface**  
495 Note 1 to entry: The details are defined in ISO/IEC 30118-1:2018.

496 **3.1.14**  
497 **Intermediary**  
498 a Device that implements both Client and Server roles and may perform protocol translation, virtual  
499 device to physical device mapping or Resource translation

500 **3.1.15**  
501 **OCF Cipher Suite**  
502 a set of algorithms and parameters that define the cryptographic functionality of a Device. The OCF  
503 Cipher Suite includes the definition of the public key group operations, signatures, and specific  
504 hashing and encoding used to support the public key.

505 **3.1.16**  
506 **OCF Cloud User – moved to OCF Cloud Security spec**

507 **3.1.17**  
508 **OCF Rooted Certificate Chain**  
509 a collection of X.509 v3 certificates in which each certificate chains to a trust anchor certificate  
510 which has been issued by a certificate authority under the direction, authority, and approval of the  
511 Open Connectivity Foundation Board of Directors as a trusted root for the OCF ecosystem.

512 **3.1.18**  
513 **Onboarding Tool (OBT)**  
514 a tool that implements DOTS(3.1.9), AMS(3.1.1) and CMS(3.1.5) functionality

515 **3.1.19**  
516 **Out of Band Communication Channel**  
517 any mechanism for delivery of a secret from one party to another, not specified by OCF

518 **3.1.20**  
519 **Owner Credential (OC)**  
520 a credential, provisioned to a Device, for the purposes of mutual authentication of the Device and  
521 OBT(3.1.18) during subsequent interactions, identified by having a Subject UUID matching the  
522 Resource Owner Id of the Device Ownership Transfer Resource hosted by a Device that has the  
523 credential

524 **3.1.21**  
525 **Platform ID**  
526 Note 1 to entry: The details are defined in ISO/IEC 30118-1:2018.

527 **3.1.22**  
528 **Property**  
529 Note 1 to entry: The details are defined in ISO/IEC 30118-1:2018.

530 **3.1.23**  
531 **Resource**  
532 Note 1 to entry: The details are defined in ISO/IEC 30118-1:2018.

533 **3.1.24**  
534 **Role (Network context)**  
535 stereotyped behavior of a Device; one of [Client, Server or Intermediary]

536 **3.1.25**  
537 **Role Identifier**  
538 a Property of an OCF credentials Resource or element in a role certificate that identifies a privileged  
539 role that a Server Device associates with a Client Device for the purposes of making authorization  
540 decisions when the Client Device requests access to Device Resources.

541 **3.1.26**  
542 **Secure Resource Manager (SRM)**  
543 a module in the OCF Core that implements security functionality that includes management of  
544 security Resources such as ACLs, credentials and Device owner transfer state.

545 **3.1.27**  
546 **Security Virtual Resource (SVR)**  
547 a resource supporting security features.

548 Note 1 to entry: For a list of all the SVRs please see clause 13.

549 **3.1.28**  
550 **Server**

551 Note 1 to entry: The details are defined in ISO/IEC 30118-1:2018.

552 **3.1.29**  
553 **Trust Anchor**

554 a well-defined, shared authority, within a trust hierarchy, by which two cryptographic entities (e.g.  
555 a Device and an OBT(3.1.18)) can assume trust

556 **3.1.30**  
557 **Device Configuration Resource (DCR)**  
558 a Resource that is any of the following:

- 559 a) a Discovery Core Resource, or
- 560 b) a Security Virtual Resource, or
- 561 c) a Wi-Fi Easy Setup Resource ("oic.r.easyssetup", "oic.r.wificonf", "oic.r.devconf"), or
- 562 d) a CoAP Cloud Configuration Resource ("oic.r.coapcloudconf"), or
- 563 e) a Software Update Resource ("oic.r.softwareupdate"), or
- 564 f) a Maintenance Resource ("oic.wk.mnt").

565 **3.1.31**  
566 **Non-Configuration Resource (NCR)**  
567 a Resource that is not a Device Configuration Resource (3.1.30).

568 **3.1.32**  
569 **Bridged Device**

570 Note 1 to entry: The details are defined in ISO/IEC 30118-3:2018.

571 **3.1.33**  
572 **Bridged Protocol**

573 Note 1 to entry: The details are defined in ISO/IEC 30118-3:2018.

574 **3.1.34**  
575 **Bridge**

576 Note 1 to entry: The details are defined in ISO/IEC 30118-3:2018.

577 **3.1.35**  
578 **Bridging Platform**

579 Note 1 to entry: The details are defined in ISO/IEC 30118-3:2018.



580 **3.1.36**  
581 **Virtual Bridged Device**  
582 Note 1 to entry: The details are defined in ISO/IEC 30118-3:2018.

583 **3.1.37**  
584 **Virtual OCF Device**  
585 Note 1 to entry: The details are defined in ISO/IEC 30118-3:2018.

586 **3.1.38**  
587 **OCF Security Domain**  
588 set of onboarded OCF Devices that are provisioned with credentialing information for confidential  
589 communication with one another

590 **3.1.39**  
591 **Owned (or "in Owned State")**  
592 having the "owned" Property of the "/oic/sec/doxm" resource equal to "TRUE"

593 **3.1.40**  
594 **Unowned (or "in Unowned State")**  
595 having the "owned" Property of the "/oic/sec/doxm" resource equal to "FALSE"

596 **3.1.41 OCF Onboarding**  
597 initial establishment of ownership over a Device, and initial provisioning of the Device for normal  
598 operation

599 **3.2 Abbreviated terms**

600 **3.2.1**  
601 **AC**  
602 Access Control

603 **3.2.2**  
604 **ACE**  
605 Access Control Entry

606 **3.2.3**  
607 **ACL**  
608 Access Control List

609 **3.2.4**  
610 **AES**  
611 Advanced Encryption Standard  
612 Note 1 to entry: See NIST FIPS 197, "Advanced Encryption Standard (AES)"

613 **3.2.5**  
614 **AMS**  
615 Access Management Service

616 **3.2.6**  
617 **CMS**  
618 Credential Management Service

619 **3.2.7**  
620 **CRUDN**  
621 CREATE, RETREIVE, UPDATE, DELETE, NOTIFY

622 **3.2.8**  
623 **CSR**  
624 Certificate Signing Request

625 **3.2.9**  
626 **CVC**  
627 Code Verification Certificate

628 **3.2.10**  
629 **ECC**  
630 Elliptic Curve Cryptography

631 **3.2.11**  
632 **ECDSA**  
633 Elliptic Curve Digital Signature Algorithm

634 **3.2.12**  
635 **EKU**  
636 Extended Key Usage

637 **3.2.13**  
638 **DOTS**  
639 Device Ownership Transfer Service

640 **3.2.14**  
641 **ID**  
642 Identity/Identifier

643 **3.2.15**  
644 **JSON**  
645 JavaScript Object Notation.

646 Note 1 to entry: See ISO/IEC 30118-1:2018.

647 **3.2.16**  
648 **JWS**  
649 JSON Web Signature.

650 Note 1 to entry: See IETF RFC 7515, "JSON Web Signature (JWS)"

651 **3.2.17**  
652 **KDF**  
653 Key Derivation Function

654 **3.2.18**  
655 **MAC**  
656 Message Authentication Code

657 **3.2.19**  
658 **MITM**  
659 Man-in-the-Middle

660 **3.2.20**  
661 **NVRAM**  
662 Non-Volatile Random-Access Memory

663 **3.2.21**  
664 **OC**  
665 Owner Credential

666 **3.2.22**  
667 **OCSP**  
668 Online Certificate Status Protocol

669 **3.2.23**  
670 **OBT**  
671 Onboarding Tool

672 **3.2.24**  
673 **OID**  
674 Object Identifier

675 **3.2.25**  
676 **OTM**  
677 Owner Transfer Method

678 **3.2.26**  
679 **OWASP**  
680 Open Web Application Security Project.

681 Note 1 to entry: See <https://www.owasp.org/>

682 **3.2.27**  
683 **PE**  
684 Policy Engine

685 **3.2.28**  
686 **PIN**  
687 Personal Identification Number

688 **3.2.29**  
689 **PPSK**  
690 PIN-authenticated pre-shared key

691 **3.2.30**  
692 **PRF**  
693 Pseudo Random Function

694 **3.2.31**  
695 **PSI**  
696 Persistent Storage Interface

697 **3.2.32**  
698 **PSK**  
699 Pre Shared Key

700 **3.2.33**  
701 **RBAC**  
702 Role Based Access Control

703 **3.2.34**  
704 **RM**  
705 Resource Manager

706 **3.2.35**  
707 **RNG**  
708 Random Number Generator

709 **3.2.36**  
710 **SBAC**  
711 Subject Based Access Control

712 **3.2.37**  
713 **SEE**  
714 Secure Execution Environment

715 **3.2.38**  
716 **SRM**  
717 Secure Resource Manager

718 **3.2.39**  
719 **SVR**  
720 Security Virtual Resource

721 **3.2.40**  
722 **SW**  
723 Software

724 **3.2.41**

725 **3.2.42**  
726 **URI**  
727 Uniform Resource Identifier

728 Note 1 to entry: See ISO/IEC 30118-1:2018.

729 **3.2.43**  
730 **VOD**  
731 Virtual OCF Device

732 Note 1 to entry: See ISO/IEC 30118-3:2018.

733 **3.2.44**  
734 **RFNOP**  
735 Ready for Normal

736 **3.2.45**  
737 **RFOTM**  
738 Ready for OTM

739 **3.2.46**  
740 **RFPRO**  
741 Ready for Provisioning

742 **3.2.47**  
743 **SRESET**  
744 Soft Reset

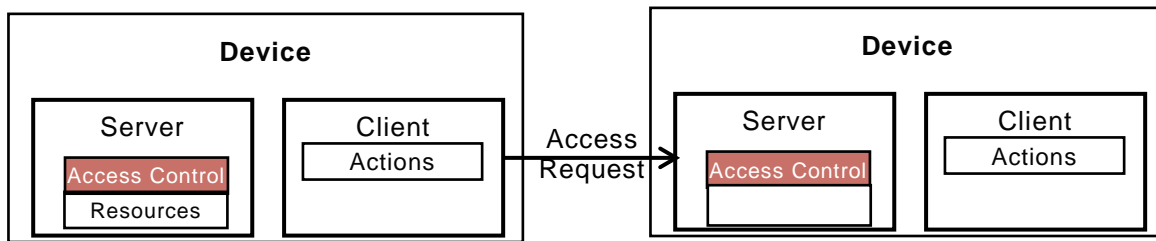
## 745 **4 Document Conventions and Organization**

### 746 **4.1 Conventions**

747 This document defines Resources, protocols and conventions used to implement security for OCF  
748 core framework and applications.

749 For the purposes of this document, the terms and definitions given in ISO/IEC 30118-1:2018 apply.

750 Figure 1 depicts interaction between OCF Devices.



751

752

**Figure 1 – OCF Interaction**

753 Devices may implement a Client role that performs Actions on Servers. Actions access Resources  
 754 managed by Servers. The OCF stack enforces access policies on Resources. End-to-end Device  
 755 interaction can be protected using session protection protocol (e.g. DTLS) or with data encryption  
 756 methods.

757 **4.2 Notation**

758 In this document, features are described as required, recommended, allowed or DEPRECATED as  
 759 follows:

760 **Required (or shall or mandatory).**

761 These basic features shall be implemented to comply with OCF Core Architecture. The phrases  
 762 "shall not", and "PROHIBITED" indicate behaviour that is prohibited, i.e. that if performed means  
 763 the implementation is not in compliance.

764 **Recommended (or should).**

765 These features add functionality supported by OCF Core Architecture and should be implemented.  
 766 Recommended features take advantage of the capabilities OCF Core Architecture, usually without  
 767 imposing major increase of complexity. Notice that for compliance testing, if a recommended  
 768 feature is implemented, it shall meet the specified requirements to be in compliance with these  
 769 guidelines. Some recommended features could become requirements in the future. The phrase  
 770 "should not" indicates behaviour that is permitted but not recommended.

771 **Allowed (may or allowed).**

772 These features are neither required nor recommended by OCF Core Architecture, but if the feature  
 773 is implemented, it shall meet the specified requirements to be in compliance with these guidelines.

774 **Conditionally allowed (CA)**

775 The definition or behaviour depends on a condition. If the specified condition is met, then the  
 776 definition or behaviour is allowed, otherwise it is not allowed.

777 **Conditionally required (CR)**

778 The definition or behaviour depends on a condition. If the specified condition is met, then the  
 779 definition or behaviour is required. Otherwise the definition or behaviour is allowed as default  
 780 unless specifically defined as not allowed.

781 **DEPRECATED**

782 Although these features are still described in this document, they should not be implemented except  
 783 for backward compatibility. The occurrence of a deprecated feature during operation of an

784 implementation compliant with the current document has no effect on the implementation's  
785 operation and does not produce any error conditions. Backward compatibility may require that a  
786 feature is implemented and functions as specified but it shall never be used by implementations  
787 compliant with this document.

788 Strings that are to be taken literally are enclosed in "double quotes".

789 Words that are emphasized are printed in *italic*.

#### 790 **4.3 Data types**

791 See ISO/IEC 30118-1:2018.

#### 792 **4.4 Document structure**

793 Informative clauses may be found in the Overview clauses, while normative clauses fall outside of  
794 those clauses.

795 The Security Specification may use the OpenAPI specification as the API definition language. The  
796 mapping of the CRUDN actions is specified in ISO/IEC 30118-1:2018.

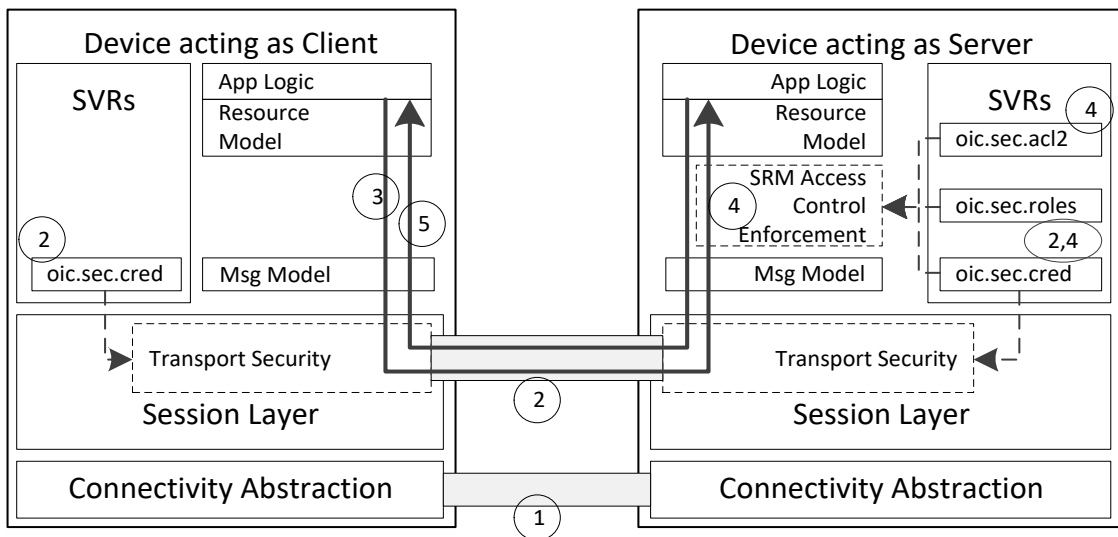
797

798 **5 Security Overview**

799 **5.1 Preamble**

800 This is an informative clause. The goal for the OCF security architecture is to protect the Resources  
 801 and all aspects of HW and SW that are used to support the protection of Resource. From OCF  
 802 perspective, a Device is a logical entity that conforms to the OCF documents. In an interaction  
 803 between the Devices, the Device acting as the Server holds and controls the Resources and  
 804 provides the Device acting as a Client with access to those Resources, subject to a set of security  
 805 mechanisms. The Platform, hosting the Device may provide security hardening that will be required  
 806 for ensuring robustness of the variety of operations described in this document.

807 The security theory of operation is depicted in Figure 2 and described in the following steps.



808

809

**Figure 2 – OCF Layers**

- 810 1) The Client establishes a network connection to the Server (Device holding the Resources). The  
 811 connectivity abstraction layer ensures the Devices are able to connect despite differences in  
 812 connectivity options.
- 813 2) The Devices (e.g. Server and Client) exchange messages either with or without a mutually-  
 814 authenticated secure channel between the two Devices.
- 815 a) The "/oic/sec/cred" Resource on each Devices holds the credentials used for mutual  
 816 authentication and (when applicable) certificate validation.
- 817 b) Messages received over a secured channel are associated with a "deviceUUID". In the case  
 818 of a certificate credential, the "deviceUUID" is in the certificate received from the other  
 819 Device. In the case of a symmetric key credential, the "deviceUUID" is configured with the  
 820 credential in the "/oic/sec/cred" Resource.
- 821 c) The Server can associate the Client with any number of roleid. In the case of mutual  
 822 authentication using a certificate, the roleid (if any) are provided in role certificates; these  
 823 are configured by the Client to the Server. In the case of a symmetric key, the allowed roleid  
 824 (if any) are configured with the credential in the "/oic/sec/cred" Resource.

825 d) Requests received by a Server over an unsecured channel are treated as anonymous and  
826 not associated with any "deviceUUID" or "roleid".

827 3) The Client submits a request to the Server.

828 4) The Server receives the request.

829 a) If the request is received over an unsecured channel, the Server treats the request as  
830 anonymous and no "deviceUUID" or "roleid" are associated with the request.

831 b) If the request is received over a secure channel, then the Server associates the  
832 "deviceUUID" with the request, and the Server associates all valid roleid of the Client with  
833 the request.

834 c) The Server then consults the Access Control List (ACL), and looks for an ACL entry  
835 matching the following criteria:

836 i) The requested Resource matches a Resource reference in the ACE

837 ii) The requested operation is permitted by the "permissions" of the ACE, and

838 iii) The "subjectUUID" contains either one of a special set of wildcard values or, if the  
839 Device is not anonymous, the subject matches the Client Deviceid associated with the  
840 request or a valid "roleid" associated with the request. The wildcard values match either  
841 all Devices communicating over an authenticated and encrypted session, or all Devices  
842 communicating over an unauthenticated and unencrypted session.

843 If there is a matching ACE, then access to the Resource is permitted; otherwise access  
844 is denied. Access is enforced by the Server's Secure Resource manager (SRM).

845 5) The Server sends a response back to the Client.

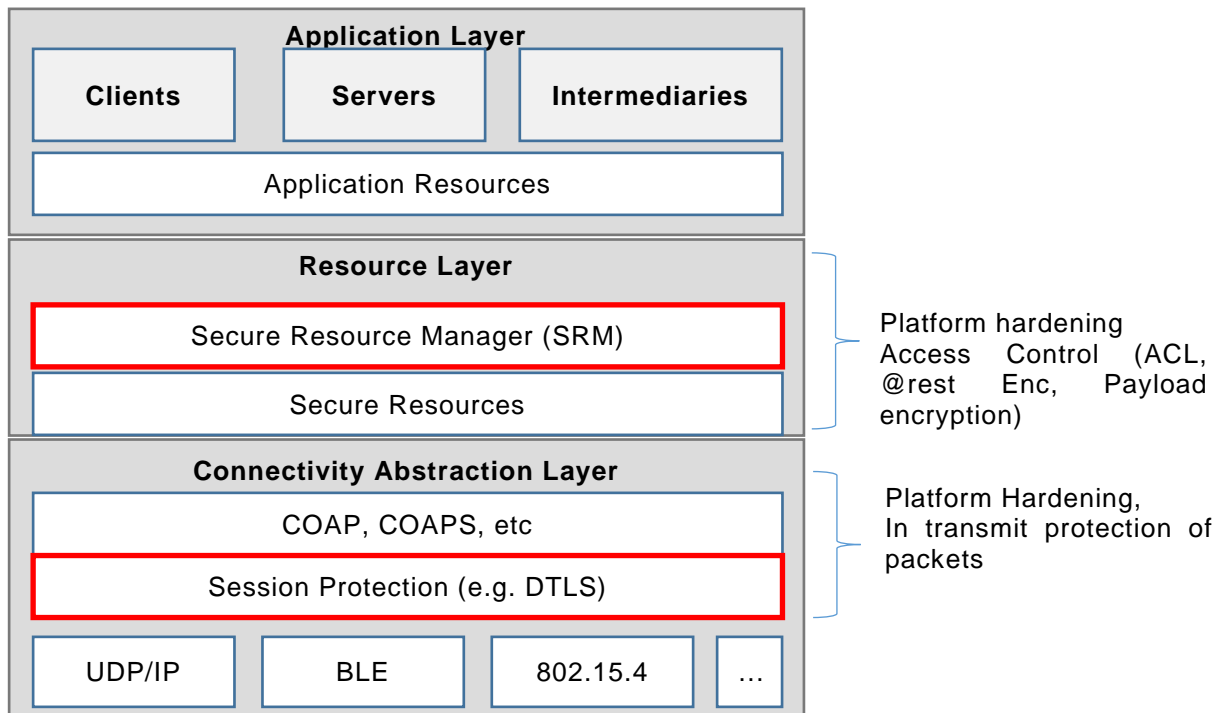
846 Resource protection includes protection of data both while at rest and during transit. Aside from  
847 access control mechanisms, the OCF Security Specification does not include specification of  
848 secure storage of Resources, while stored at Servers. However, at rest protection for security  
849 Resources is expected to be provided through a combination of secure storage and access control.  
850 Secure storage can be accomplished through use of hardware security or encryption of data at rest.  
851 The exact implementation of secure storage is subject to a set of hardening requirements that are  
852 specified in clause 14 and may be subject to certification guidelines.

853 Data in transit protection, on the other hand, will be specified fully as a normative part of this  
854 document. In transit protection may be afforded at the resource layer or transport layer. This  
855 document only supports in transit protection at transport layer through use of mechanisms such as  
856 DTLS.

857 NOTE: DTLS will provide packet by packet protection, rather than protection for the payload as whole. For instance, if  
858 the integrity of the entire payload as a whole is required, separate signature mechanisms must have already been in  
859 place before passing the packet down to the transport layer.

860 Figure 3 depicts OCF Security Enforcement Points.





861  
862  
863 **Figure 3 – OCF Security Enforcement Points**

864 **5.2 Access Control**

865 The OCF framework assumes that Resources are hosted by a Server and are made available to  
866 Clients subject to access control and authorization mechanisms. The Resources at the end point  
867 are protected through implementation of access control, authentication and confidentiality  
868 protection. This clause provides an overview of Access Control (AC) through the use of ACLs.  
869 However, AC in the OCF stack is expected to be transport and connectivity abstraction layer  
870 agnostic.

871 Implementation of access control relies on a-priori definition of a set of access policies for the  
872 Resource. The policies may be stored by a local ACL or an Access Management Service (AMS) in  
873 form of Access Control Entries (ACE). Two types of access control mechanisms can be applied:

- 874 – Subject-based access control (SBAC), where each ACE will match a subject (e.g. identity of  
875 requestor) of the requesting entity against the subject included in the policy defined for  
876 Resource. Asserting the identity of the requestor requires an authentication process.
- 877 – Role-based Access Control (RBAC), where each ACE will match a role identifier included in the  
878 policy for the Resource to a role identifier associated with the requestor.

879 Some Resources, such as Collections, generate requests to linked Resources when appropriate  
880 Interfaces are used. In such cases, additional access control considerations are necessary.  
881 Additional access control considerations for Collections when using the batch OCF Interface are  
882 found in clause 12.2.7.3.

883 In the OCF access control model, access to a Resource instance requires an associated ACE. The  
884 lack of such an associated ACE results in the Resource being inaccessible.

885 The ACE only applies if the ACE matches both the subject (i.e. OCF Client) and the requested  
886 Resource. There are multiple ways a subject could be matched, (1) DeviceID, (2) Role Identifier or  
887 (3) wildcard. The way in which the client connects to the server may be relevant context for making

888 access control decisions. Wildcard matching on authenticated vs. unauthenticated and encrypted  
889 vs. unencrypted connection allows an access policy to be broadly applied to subject classes.

890 Example Wildcard Matching Policy:

```
891 "aclist2": [  
892   {  
893     "subject": {"conntype": "anon-clear" },  
894     "resources": [  
895       { "wc": "*" }  
896     ],  
897     "permission": 31  
898   },  
899   {  
900     "subject": {"conntype": "auth-crypt" },  
901     "resources": [  
902       { "wc": "*" }  
903     ],  
904     "permission": 31  
905   },  
906 ]
```

907 Details of the format for ACL are defined in clause 12. The ACL is composed of one or more ACEs.  
908 The ACL defines the access control policy for the Devices.

909 ACL Resource requires the same security protection as other sensitive Resources, when it comes  
910 to both storage and handling by SRM and PSI. Thus hardening of an underlying Platform (HW and  
911 SW) must be considered for protection of ACLs and as explained in clause 5.2.2 ACLs may have  
912 different scoping levels and thus hardening needs to be specially considered for each scoping level.  
913 For instance, a physical device may host multiple Device implementations and thus secure storage,  
914 usage and isolation of ACLs for different Servers on the same Device needs to be considered.

## 915 **5.2.1 ACL Architecture**

### 916 **5.2.1.1 ACL Architecture General**

917 The Server examines the Resource(s) requested by the client before processing the request. The  
918 access control resource is searched to find one or more ACE entries that match the requestor and  
919 the requested Resources. If a match is found, then permission and period constraints are applied.  
920 If more than one match is found, then the logical UNION of permissions is applied to the overlapping  
921 periods.

922 The server uses the connection context to determine whether the subject has authenticated or not  
923 and whether data confidentiality has been applied or not. Subject matching wildcard policies can  
924 match on each aspect. If the user has authenticated, then subject matching may happen at  
925 increased granularity based on role or device identity.

926 Each ACE contains the permission set that will be applied for a given Resource requestor.  
927 Permissions consist of a combination of CREATE, RETREIVE, UPDATE, DELETE and NOTIFY  
928 (CRUDN) actions. Requestors authenticate as a Device and optionally operating with one or more  
929 roles. Devices may acquire elevated access permissions when asserting a role. For example, an  
930 ADMINISTRATOR role might expose additional Resources and OCF Interfaces not normally  
931 accessible.

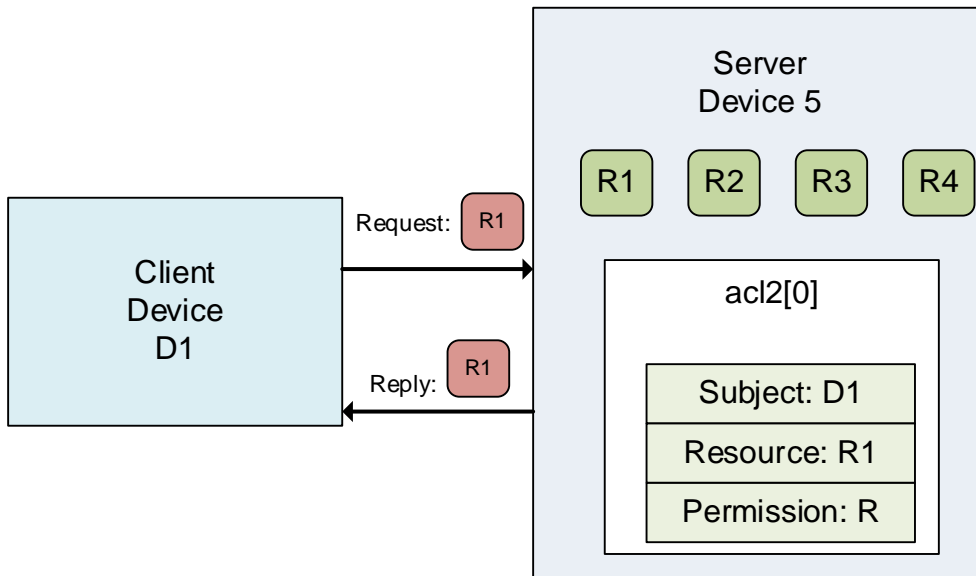
932 **5.2.1.2 Use of local ACLs**

933 Servers may host ACL Resources locally. Local ACLs allow greater autonomy in access control  
934 processing than remote ACL processing by an AMS.

935 The following use cases describe the operation of access control

936 Use Case 1: As depicted in Figure 4, Server Device hosts 4 Resources (R1, R2, R3 and R4). Client  
937 Device D1 requests access to Resource R1 hosted at Server Device 5. ACL[0] corresponds to  
938 Resource R1 and includes D1 as an authorized subject. Thus, Device D1 receives access to  
939 Resource R1 because the local ACL "/oic/sec/acl2/0" matches the request.

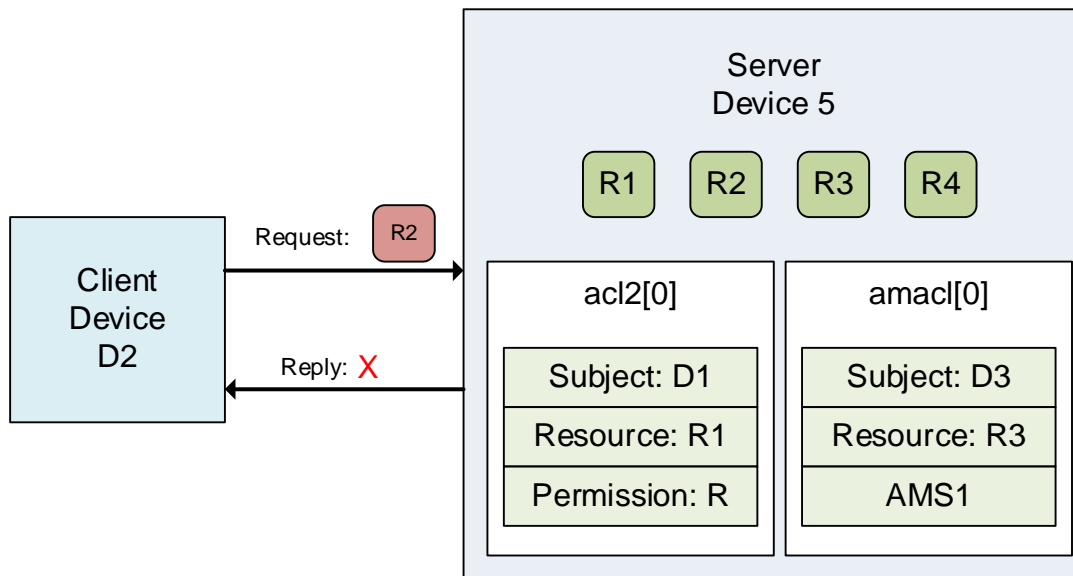
940



941

942 **Figure 4 – Use case-1 showing simple ACL enforcement**

943 Use Case 2: As depicted in Figure 5, Client Device D2 access is denied because no local ACL  
944 match is found for subject D2 pertaining Resource R2 and no AMS policy is found.



946

947

**Figure 5 – Use case 2: A policy for the requested Resource is missing**

### 948 5.2.1.3 Use of AMS

949 AMS improves ACL policy management. However, they can become a central point of failure. Due to network latency overhead, ACL processing may be slower through an AMS.  
950

951 AMS centralizes access control decisions, but Server Devices retain enforcement duties.

952 The AMS is authenticated by referencing a credential issued to the device identifier contained in  
953 `"/oic/sec/acl2.rownruuid"`.

### 954 5.2.2 Access Control Scoping Levels

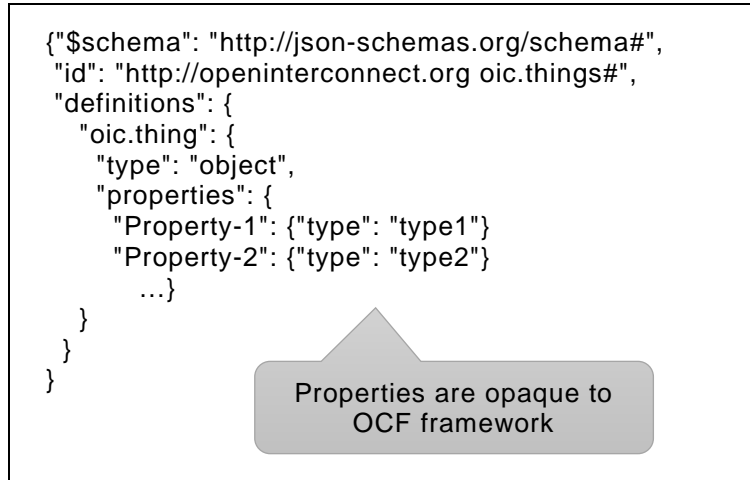
955 **Group Level Access** - Group scope means applying AC to the group of Devices that are grouped  
956 for a specific context. Group Level Access means all group members have access to group data  
957 but non-group members must be granted explicit access. Group level access is implemented using  
958 Role Credentials and/or connection type

959 **OCF Device Level Access** – OCF Device scope means applying AC to an individual Device, which  
960 may contain multiple Resources. Device level access implies accessibility extends to all Resources  
961 available to the Device identified by Device ID. Credentials used for AC mechanisms at Device are  
962 OCF Device-specific.

963 **OCF Resource Level Access** – OCF Resource level scope means applying AC to individual  
964 Resources. Resource access requires an ACL that specifies how the entity holding the Resource  
965 (Server) shall make a decision on allowing a requesting entity (Client) to access the Resource.

966 **Property Level Access** - Property level scope means applying AC only to an individual Property.  
967 Property level access control is only achieved by creating a Resource that contains a single  
968 Property.

969 Controlling access to static Resources where it is impractical to redesign the Resource, it may  
 970 appropriate to introduce a collection Resource that references the child Resources having separate  
 971 access permissions. An example is shown Figure 6, where an "oic.thing" Resource has two  
 972 properties: Property-1 and Property-2 that would require different permissions.

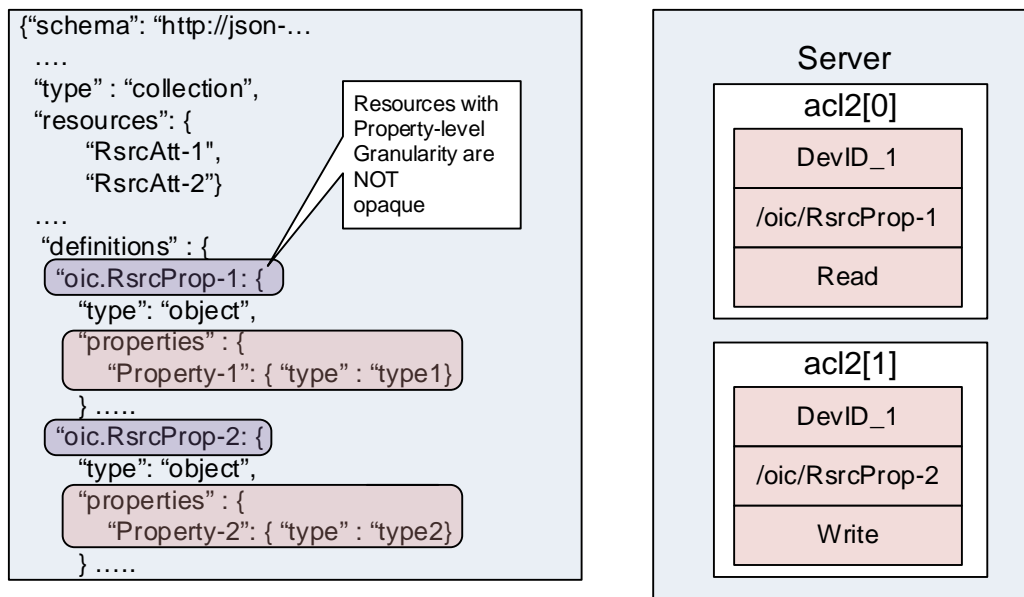


973

974

**Figure 6 – Example Resource definition with opaque Properties**

975 Currently, OCF framework treats property level information as opaque; therefore, different  
 976 permissions cannot be assigned as part of an ACL policy (e.g. read-only permission to Property-1  
 977 and write-only permission to Property-2). Thus, as shown in Figure 7, the "oic.thing" is split into  
 978 two new Resource "oic.RsrcProp-1" and "oic.RsrcProp-2". This way, Property level ACL can be  
 979 achieved through use of Resource-level ACLs.



980

981

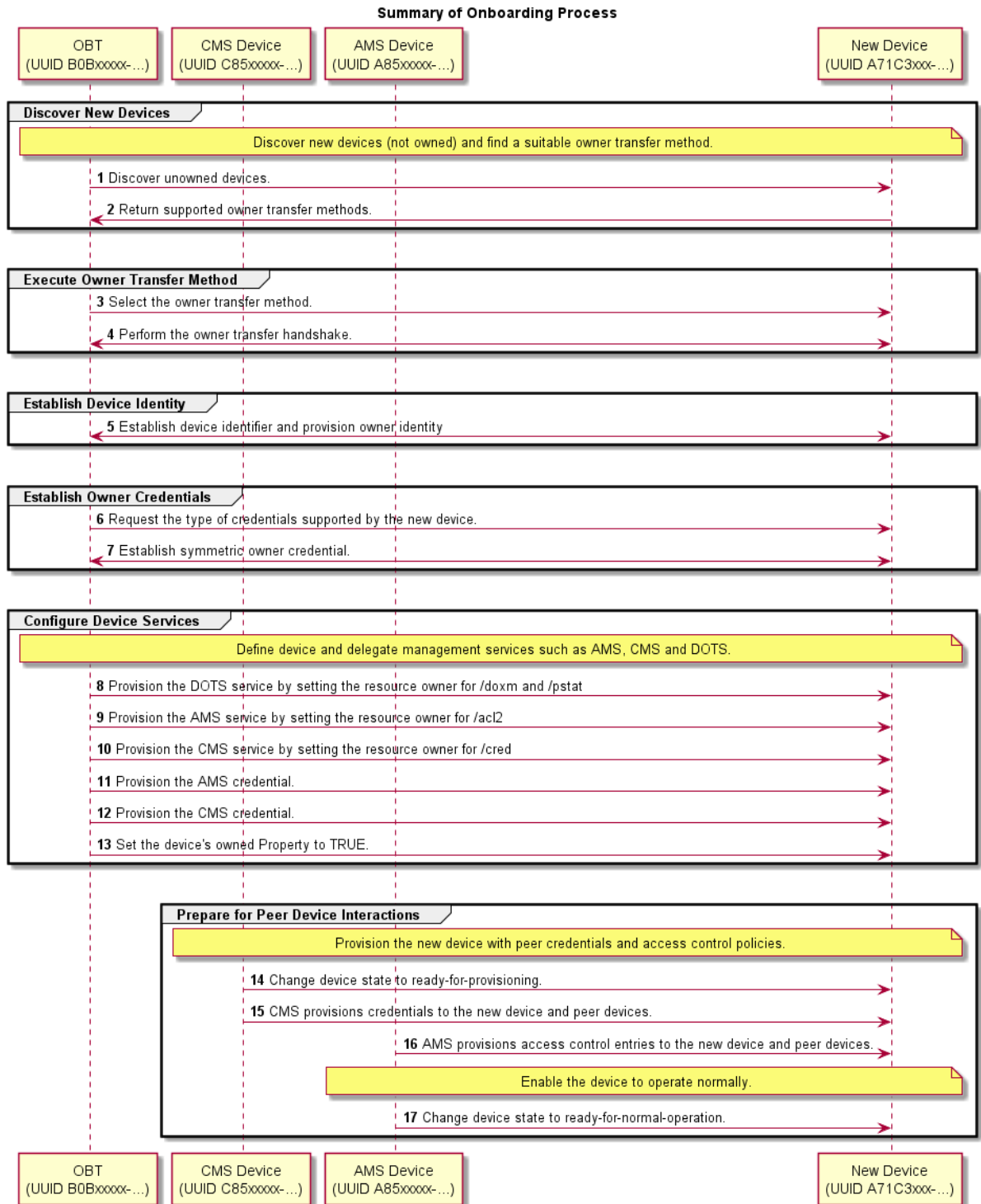
**Figure 7 – Property Level Access Control**

982 **5.3 Onboarding Overview**

983 **5.3.1 Onboarding General**

984 Before a Device becomes operational in an OCF environment and is able to interact with other  
985 Devices, it needs to be appropriately onboarded. The first step in onboarding a Device is to  
986 configure the ownership where the legitimate user that owns/purchases the Device uses an  
987 Onboarding tool (OBT) and using the OBT uses one of the Owner Transfer Methods (OTMs) to  
988 establish ownership. Once ownership is established, the OBT provisions the Device, at the end of  
989 which the Device becomes operational and is able to interact with other Devices in an OCF  
990 environment.

991 Figure 8 depicts Onboarding Overview.



**Figure 8 – Onboarding Overview**

992  
993

994 This clause explains the onboarding and security provisioning process but leaves the provisioning  
 995 of non-security aspects to other OCF documents. In the context of security, all Devices are required  
 996 to be provisioned with minimal security configuration that allows the Device to securely  
 997 interact/communicate with other Devices in an OCF environment. This minimal security

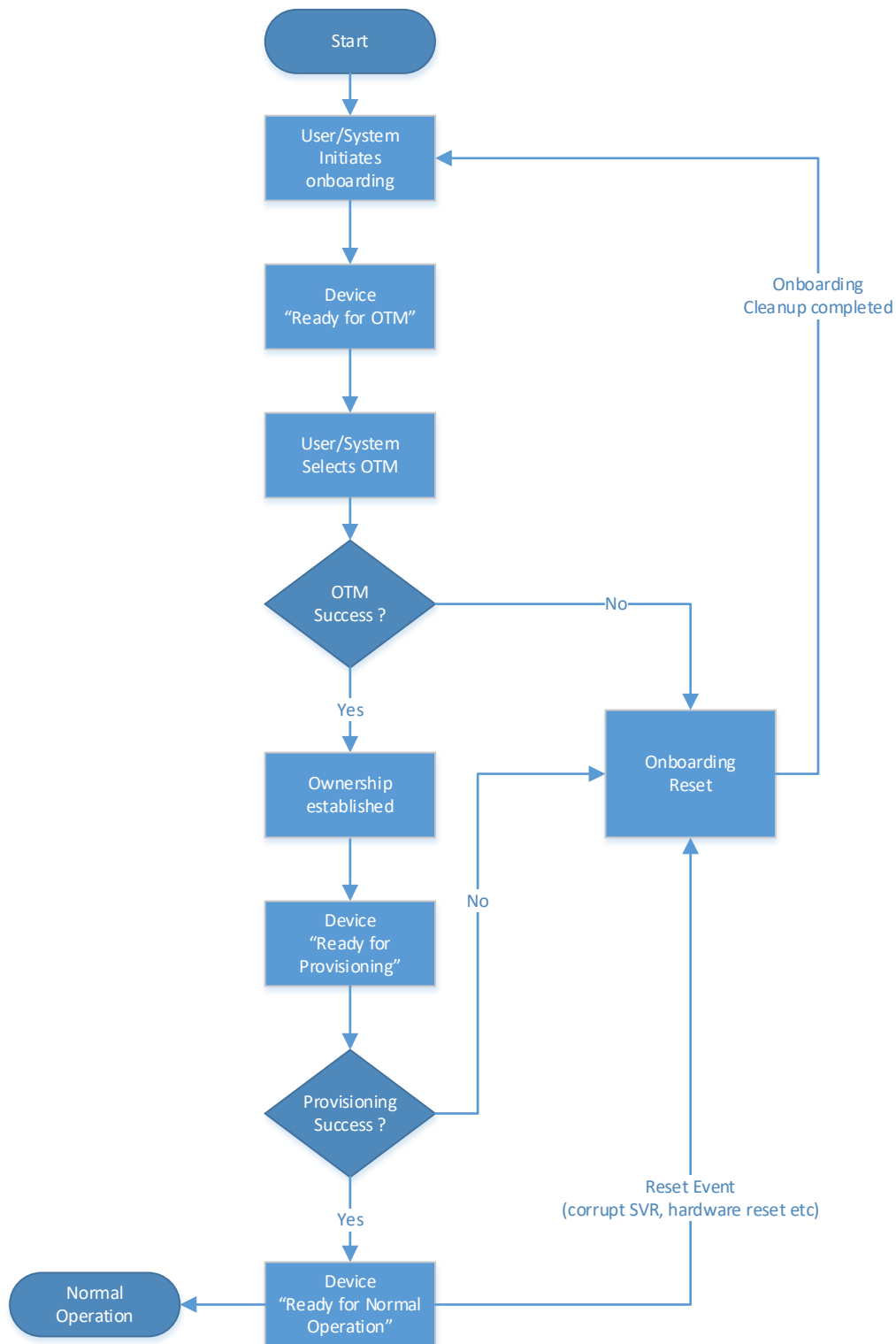
998 configuration is defined as the Onboarded Device "Ready for Normal Operation" and is specified  
999 in 7.5.

1000 Onboarding and provisioning implementations could utilize services defined outside this document,  
1001 it is expected that in using other services, trust between the device being onboarded and the  
1002 various tools is not transitive. This implies that the device being onboarded will individually  
1003 authenticate the credentials of each and every tool used during the onboarding process; that the  
1004 tools not share credentials or imply a trust relationship where one has not been established.

### 1005 **5.3.2 Onboarding Steps**

1006 The flowchart in Figure 9 shows the typical steps that are involved during onboarding. Although  
1007 onboarding may include a variety of non-security related steps, the diagram focus is mainly on the  
1008 security related configuration to allow a new Device to function within an OCF environment.  
1009 Onboarding typically begins with the Device becoming an Owned Device followed by configuring  
1010 the Device for the environment that it will operate in. This would include setting information such  
1011 as who can access the Device and what actions can be performed as well as what permissions the  
1012 Device has for interacting with other Devices.





1013

1014

**Figure 9 – OCF Onboarding Process**

1015

### 5.3.3 Establishing a Device Owner

1016

The objective behind establishing Device ownership is to allow the legitimate user that owns/purchased the Device to assert itself as the owner and manager of the Device. This is done

1017

1018 through the use of a DOTS that includes the creation of an ownership context between the new  
1019 Device and the DOTS and asserts operational control and management of the Device. The DOTS  
1020 is hosted on an OBT.

1021 The DOTS uses one of the OTMs specified in 7.3 to securely establish Device ownership.

1022 An OTM establishes a new owner (the operator of DOTS) that is authorized to manage the Device.  
1023 Owner transfer establishes the following

1024 – The DOTS provisions an Owner Credential (OC) to the "creds" Property in the "/oic/sec/cred"  
1025 Resource of the Device. This OC allows the Device and DOTS to mutually authenticate during  
1026 subsequent interactions. The OC associates the DOTS Device UUID with the "rowneruid"  
1027 Property of the "/oic/sec/doxm" Resource establishing it as the resource owner.

1028 – The Device owner establishes trust in the Device through the OTM.

1029 – Preparing the Device for provisioning by providing credentials that may be needed.

#### 1030 **5.3.4 Provisioning for Normal Operation**

1031 Once the Device has the necessary information to initiate provisioning, the next step is to provision  
1032 additional security configuration that allows the Device to become operational. This can include  
1033 setting various parameters and may also involve multiple steps. Also provisioning of ACL's for the  
1034 various Resources hosted by the Server on the Device is done at this time. The provisioning step  
1035 is not limited to this stage only. Device provisioning can happen at multiple stages in the Device's  
1036 operational lifecycle. However specific security related provisioning of Resource and Property state  
1037 would likely happen at this stage at the end of which, each Device reaches the Onboarded Device  
1038 "Ready for Normal Operation" State. The "Ready for Normal Operation" State is expected to be  
1039 consistent and well defined regardless of the specific OTM used or regardless of the variability in  
1040 what gets provisioned. However individual OTM mechanisms and provisioning steps may specify  
1041 additional configuration of Resources and Property states. The minimal mandatory configuration  
1042 required for a Device to be in "Ready for Normal Operation" state is specified in 8.

#### 1043 **5.3.5 Device Provisioning for OCF Cloud and Device Registration Overview – moved to** 1044 **OCF Cloud Security document**

1045 This clause is intentionally left blank.

#### 1046 **5.3.6 OCF Compliance Management System**

1047 The OCF Compliance Management System (OCMS) is a service maintained by the OCF that  
1048 provides Certification status and information for OCF Devices.

1049 The OCMS shall provide a JSON-formatted Certified Product List (CPL), hosted at the URI:  
1050 <https://www.openconnectivity.org/certification/ocms-cpl.json>

1051 The OBT shall possess the Root Certificate needed to enable https connection to the URI  
1052 <https://www.openconnectivity.org/certification/ocms-cpl.json>.

1053 The OBT should periodically refresh its copy of the CPL via the URI  
1054 <https://www.openconnectivity.org/certification/ocms-cpl.json>, as appropriate to OCF Security  
1055 Domain owner policy requirements.

### 1056 **5.4 Provisioning**

#### 1057 **5.4.1 Provisioning General**

1058 In general, provisioning may include processes during manufacturing and distribution of the Device  
1059 as well as processes after the Device has been brought into its intended environment (parts of  
1060 onboarding process). In this document, security provisioning includes, processes after ownership  
1061 transfer (even though some activities during ownership transfer and onboarding may lead to  
1062 provisioning of some data in the Device) configuration of credentials for interacting with  
Copyright Open Connectivity Foundation, Inc. © 2016-2020. All rights Reserved 23

1063 provisioning services, configuration of any security related Resources and credentials for dealing  
1064 with any services that the Device need to contact later on.

1065 Once the ownership transfer is complete, the Device needs to engage with the CMS and AMS to  
1066 be provisioned with proper security credentials and parameters for regular operation. These  
1067 parameters can include:

- 1068 – Security credentials through a CMS, currently assumed to be deployed in the same OBT.
- 1069 – Access control policies and ACLs through an AMS, currently assumed to be deployed in the  
1070 same OBT, but may be part of AMS in future.

1071 Devices are aware of their security provisioning status. Self-awareness allows them to be proactive  
1072 about provisioning or re-provisioning security Resources as needed to achieve the devices  
1073 operational goals.

#### 1074 **5.4.2 Provisioning other services**

1075 To be able to support the use of potentially different device management service hosts, each Device  
1076 Secure Virtual Resource (SVR) has an associated Resource owner identified in the Resource's  
1077 rowneruuid Property.

1078 The "rowneruuid" Property of the "/oic/sec/doxm" and "/oic/sec/pstat" resources identifies the  
1079 DOTS.

1080 The "rowneruuid" Property of the "/oic/sec/cred" resource identifies the CMS.

1081 The "rowneruuid" Property of the "/oic/sec/acl2" resource identifies the AMS.

1082 The DOTS provisions credentials that enable secure connections between OCF Services and the  
1083 new Device. The DOTS initiates client-directed provisioning by signaling the OCF Service.

#### 1084 **5.4.3 Provisioning Credentials for Normal Operation**

1085 The "/oic/sec/cred" Resource supports multiple types of credentials including:

- 1086 – Pairwise symmetric keys
- 1087 – Group symmetric keys
- 1088 – Certificates
- 1089 – Raw asymmetric keys

1090 The CMS securely provisions credentials for Device-to-Device interactions using the CMS  
1091 credential provisioned by the DOTS.

1092 The following example describes how a Device updates a symmetric key credential involving a peer  
1093 Device. The Device discovers the credential to be updated; for example, a secure connection  
1094 attempt fails. The CMS returns an updated symmetric key credential. The CMS updates the  
1095 corresponding symmetric key credential on the peer Device.

#### 1096 **5.4.4 Role Assignment and Provisioning for Normal Operation**

1097 The Servers, receiving requests for Resources they host, need to verify the role identifier(s)  
1098 asserted by the Client requesting the Resource and compare that role identifier(s) with the  
1099 constraints described in the Server's ACLs Thus, a Client Device may need to be provisioned with  
1100 one or more role credentials.

1101 Each Device holds the role information as a Property within the credential Resource.

1102 Once provisioned, the Client can assert the role it is using as described in 10.4.2, if it has a  
1103 certificate role credential.

1104 All provisioned roles are used in ACL enforcement. When a server has multiple roles provisioned  
1105 for a client, access to a Resource is granted if it would be granted under any of the roles.

#### 1106 5.4.5 ACL provisioning

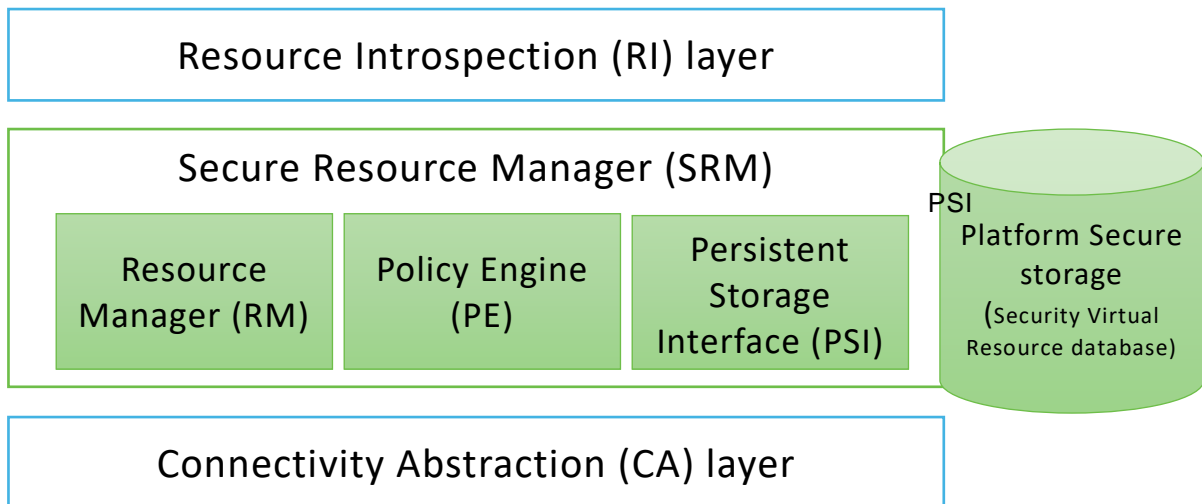
1107 ACL provisioning is performed over a secure connection between the AMS and its Devices. The  
1108 AMS provisions the ACL by updating the Device's ACL Resource.

#### 1109 5.5 Secure Resource Manager (SRM)

1110 SRM plays a key role in the overall security operation. In short, SRM performs both management  
1111 of SVR and access control for requests to access and manipulate Resources. SRM consists of 3  
1112 main functional elements:

- 1113 – A Resource manager (RM): responsible for 1) Loading SVRs from persistent storage (using PSI)  
1114 as needed. 2) Supplying the Policy Engine (PE) with Resources upon request. 3) Responding  
1115 to requests for SVRs. While the SVRs are in SRM memory, the SVRs are in a format that is  
1116 consistent with device-specific data store format. However, the RM will use JSON format to  
1117 marshal SVR data structures before being passed to PSI for storage, or travel off-device.
- 1118 – A Policy Engine (PE) that takes requests for access to SVRs and based on access control  
1119 policies responds to the requests with either "ACCESS\_GRANTED" or "ACCESS\_DENIED". To  
1120 make the access decisions, the PE consults the appropriate ACL and looks for best Access  
1121 Control Entry (ACE) that can serve the request given the subject (Device or role) that was  
1122 authenticated by DTLS.
- 1123 – Persistent Storage Interface (PSI): PSI provides a set of APIs for the RM to manipulate files in  
1124 its own memory and storage. The SRM design is modular such that it may be implemented in  
1125 the Platform's secure execution environment; if available.

1126 Figure 10 depicts OCF's SRM Architecture.



1127

1128 **Figure 10 – OCF's SRM Architecture**

#### 1129 5.6 Credential Overview

1130 Devices may use credentials to prove the identity and role(s) of the parties in bidirectional  
1131 communication. Credentials can be symmetric or asymmetric. Each device stores secret and public

1132 parts of its own credentials where applicable, as well as credentials for other devices that have  
1133 been provided by the DOTS or a CMS. These credentials are then used in the establishment of  
1134 secure communication sessions (e.g. using DTLS) to validate the identities of the participating  
1135 parties. Role credentials are used once an authenticated session is established, to assert one or  
1136 more roles for a device.

1137

## 1138 6 Security for the Discovery Process

### 1139 6.1 Preamble

1140 The main function of a discovery mechanism is to provide Universal Resource Identifiers (URIs,  
1141 called links) for the Resources hosted by the Server, complemented by attributes about those  
1142 Resources and possible further link relations. (in accordance to clause 10 in ISO/IEC 30118-1:2018)

### 1143 6.2 Security Considerations for Discovery

1144 When defining discovery process, care must be taken that only a minimum set of Resources are  
1145 exposed to the discovering entity without violating security of sensitive information or privacy  
1146 requirements of the application at hand. This includes both data included in the Resources, as well  
1147 as the corresponding metadata.

1148 To achieve extensibility and scalability, this document does not provide a mandate on  
1149 discoverability of each individual Resource. Instead, the Server holding the Resource will rely on  
1150 ACLs for each Resource to determine if the requester (the Client) is authorized to see/handle any  
1151 of the Resources.

1152 The "/oic/sec/acl2" Resource contains ACL entries governing access to the Server hosted  
1153 Resources. (See 13.5)

1154 Aside from the privacy and discoverability of Resources from ACL point of view, the discovery  
1155 process itself needs to be secured. This document sets the following requirements for the discovery  
1156 process:

- 1157 1) Providing integrity protection for discovered Resources.
- 1158 2) Providing confidentiality protection for discovered Resources that are considered sensitive.

1159 The discovery of Resources is done by doing a RETRIEVE operation (either unicast or multicast)  
1160 on the known "/oic/res" Resource.

1161 The discovery request is sent over a non-secure channel (multicast or unicast without DTLS), a  
1162 Server cannot determine the identity of the requester. In such cases, a Server that wants to  
1163 authenticate the Client before responding can list the secure discovery URI (e.g.  
1164 coaps://IP:PORT/oic/res ) in the unsecured "/oic/res" Resource response. This means the secure  
1165 discovery URI is by default discoverable by any Client. The Client will then be required to send a  
1166 separate unicast request using DTLS to the secure discovery URI.

1167 For example, a Client with Device Id "d1" (UUID:"0685B960-736F-46F7-BEC0-9E6CBD61ADC1")  
1168 makes a RETRIEVE request on the "/door" Resource hosted on a Server with Device Id "d3" where  
1169 d3 has the ACL2s:

```
1170 {  
1171   "aclist2": [  
1172     {  
1173       "subject": {"uuid": "0685B960-736F-46F7-BEC0-9E6CBD61ADC1"},  
1174       "resources": [{"href": "/door"}],  
1175       "permission": 2, // RETRIEVE  
1176       "aceid": 1  
1177     },  
1178     {  
1179       "subject": {"authority": "owner", "role": "owner"},  
1180       "resources": [{"href": "/door"}],  
1181       "permission": 2, // RETRIEVE
```

```

1182     "aceid": 2
1183   },
1184   {
1185     "subject": {"uuid": "0685B960-736F-46F7-BEC0-9E6CBD61ADC1"},
1186     "resources": [{"href": "/door/lock"}],
1187     "permission": 4, // UPDATE
1188     "aceid": 3
1189   }
1190 ],
1191 "rowneruuid": "0685B960-736F-46F7-BEC0-9E6CBD61ADC1"
1192 }

```

1193 The ACL indicates that Client "d1" has RETRIEVE permissions on the Resource. Hence when  
1194 device "d1" does a discovery on the "/door" Resource of the Server "d3", the response will include  
1195 all the URIs in the "/door" Resource. Client "d2" without a Role ID "owner" will get an error response  
1196 that includes no URI.

1197 Discovery results delivered to d1 regarding d3's "/door" Resource from the secure interface:

```

1198 [
1199   {
1200     "href": "/door",
1201     "rel": "self",
1202     "rt": ["oic.wk.col"],
1203     "if": ["oic.if.ll", "oic.if.b", "oic.if.baseline"],
1204     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:5555"}]
1205   },
1206   {
1207     "href": "/door/lock",
1208     "rt": ["oic.r.lock.status"],
1209     "if": ["oic.if.a", "oic.if.baseline"],
1210     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:5555"}]
1211   }
1212 ]

```

## 1213 **7 Security Provisioning**

### 1214 **7.1 Device Identity**

#### 1215 **7.1.1 General Device Identity**

1216 Each Device, which is a logical device, is identified with a Device ID.

1217 Devices shall be identified by a Device ID value that is established as part of device onboarding.  
1218 The "/oic/sec/doxm" Resource specifies the Device ID format (e.g. "urn:uuid"). Device IDs shall be  
1219 unique within the scope of operation of the corresponding OCF Security Domain, and should be  
1220 universally unique. The DOTS shall ensure Device ID of the new Device is unique within the scope  
1221 of the owner's OCF Security Domain. The DOTS shall verify the chosen new device identifier does  
1222 not conflict with Device IDs previously introduced into the OCF Security Domain.

1223 Devices maintain an association of Device ID and cryptographic credential using a "/oic/sec/cred"  
1224 Resource. Devices regard the "/oic/sec/cred" Resource as authoritative when verifying  
1225 authentication credentials of a peer device.

1226 A Device maintains its Device ID in the "/oic/sec/doxm" Resource. It maintains a list of credentials,  
1227 both its own and other Device credentials, in the "/oic/sec/cred" Resource. The device ID can be  
1228 used to distinguish between a device's own credential, and credentials for other devices.  
1229 Furthermore, the "/oic/sec/cred" Resource may contain multiple credentials for the device.

1230 When using manufacturer certificates, the certificate should bind the ID to the stored secret in the  
1231 device as described later in this clause.

1232 A physical Device, referred to as a Platform in OCF documents, may host multiple Devices. The  
1233 Platform is identified by a Platform ID. The Platform ID shall be globally unique and inserted in the  
1234 device in an integrity protected manner (e.g. inside secure storage or signed and verified).

1235 An OCF Platform may have a secure execution environment, which shall be used to secure unique  
1236 identifiers and secrets. If a Platform hosts multiple devices, some mechanism is needed to provide  
1237 each Device with the appropriate and separate security.

#### 1238 **7.1.2 Device Identity for Devices with UAID [Deprecated]**

1239 This clause is intentionally left blank.

### 1240 **7.2 Device Ownership**

1241 This is an informative clause. Devices are logical entities that are security endpoints that have an  
1242 identity that is authenticable using cryptographic credentials. A Device is Unowned when it is first  
1243 initialized. Establishing device ownership is a process by which the device asserts its identity to  
1244 the DOTS and the DOTS provisions an owner identity. This exchange results in the device changing  
1245 its ownership state, thereby preventing a different DOTS from asserting administrative control over  
1246 the device.

1247 The ownership transfer process starts with the OBT discovering a new device that is in Unowned  
1248 state through examination of the "Owned" Property of the "/oic/sec/doxm" Resource of the new  
1249 device. At the end of ownership transfer, the following is accomplished:

- 1250 1) The DOTS establishes a secure session with new device.
- 1251 2) Optionally asserts any of the following:
  - 1252 a) Proximity (using PIN) of the OBT to the Platform.
  - 1253 b) Manufacturer's certificate asserting Platform vendor, model and other Platform specific  
1254 attributes.
- 1255 3) Determines the device identifier.



- 1256 4) Determines the device owner.
- 1257 5) Specifies the device owner (e.g. Device ID of the OBT).
- 1258 6) Provisions the device with owner's credentials.
- 1259 7) Sets the "Owned" state of the new device to TRUE.

1260 .

1261 **7.3 Device Ownership Transfer Methods**

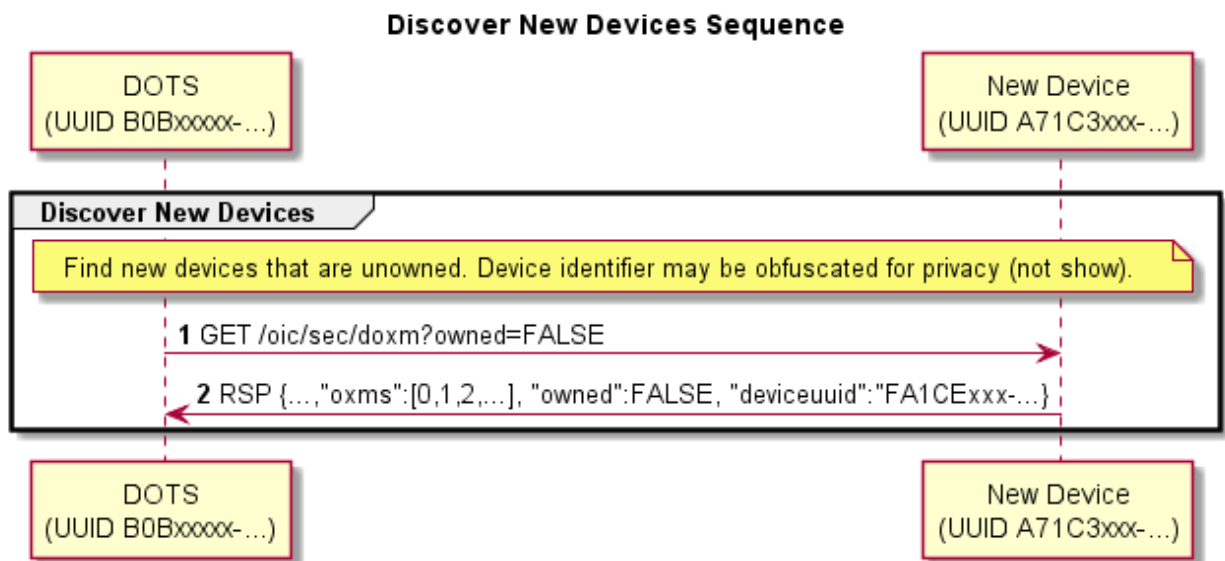
1262 **7.3.1 OTM implementation requirements**

1263 This document provides specifications for several methods for ownership transfer. Implementation  
 1264 of each individual ownership transfer method is considered optional. However, each device shall  
 1265 implement at least one of the ownership transfer methods not including vendor specific methods.

1266 All OTMs included in this document are considered optional. Each vendor is required to choose  
 1267 and implement at least one of the OTMs specified in this document. The OCF, does however,  
 1268 anticipate vendor-specific approaches will exist. Should the vendor wish to have interoperability  
 1269 between a vendor-specific OTM and OBTs from other vendors, the vendor must work directly with  
 1270 OBT vendors to ensure interoperability. Notwithstanding, standardization of OTMs is the preferred  
 1271 approach. In such cases, a set of guidelines is provided in 7.3.7 to help vendors in designing  
 1272 vendor-specific OTMs.

1273 The "/oic/sec/doxm" Resource is extensible to accommodate vendor-defined owner transfer  
 1274 methods (OTM). The DOTS determines which OTM is most appropriate to onboard the new Device.  
 1275 All OTMs shall represent the onboarding capabilities of the Device using the "oxms" Property of  
 1276 the "/oic/sec/doxm" Resource. The DOTS queries the Device's supported credential types using  
 1277 the "credtype" Property of the "/oic/sec/cred" Resource. The DOTS and CMS provision credentials  
 1278 according to the credential types supported.

1279 Figure 11 depicts new Device discovery sequence.



1280

1281 **Figure 11 – Discover New Device Sequence**

1282

**Table 1 – Discover New Device Details**

Step	Description
1	The DOTS queries to see if the new device is not yet owned.
2	The new device returns the "/oic/sec/doxm" Resource containing ownership status and supported OTMs. It also contains a temporal device ID that may change subsequent to successful owner transfer. The device should supply a temporal ID to facilitate discovery as a guest device. Clause 7.3.9 provides security considerations regarding selecting an OTM.

1284 Vendor-specific device OTMs shall adhere to the "/oic/sec/doxm" Resource Specification for OCS  
1285 that results from vendor-specific device OTM. Vendor-specific OTM should include provisions for  
1286 establishing trust in the new Device by the DOTS and optionally establishing trust in the OBT by  
1287 the new Device.

1288 The new device may have to perform some initialization steps at the beginning of an OTM. For  
1289 example, if the Random PIN Based OTM is initiated, the new device may generate a random PIN  
1290 value. The DOTS updates the oxmsel property of "/oic/sec/doxm" to the value corresponding to the  
1291 OTM being used, before performing other OTM steps. This update notifies the new device that  
1292 ownership transfer is starting.

1293 The end state of a vendor-specific OTM shall allow the new Device to authenticate to the OBT and  
1294 the OBT to authenticate to the new device.

1295 Additional provisioning steps may be performed subsequent to owner transfer success leveraging  
1296 the established OTM session.

### 1297 **7.3.2 SharedKey Credential Calculation**

1298 The SharedKey credential is derived using a PRF that accepts the key\_block value resulting from  
1299 the DTLS handshake used for onboarding. The new Device shall use the following calculation to  
1300 ensure interoperability across vendor products (the DOTS performs the same calculation):

1301 SharedKey = *PRF*(Secret, Message);

1302 Where:

- 1303 - PRF shall use TLS 1.2 PRF defined by IETF RFC 5246 clause 5.
- 1304 - Secret is the key\_block resulting from the DTLS handshake
  - 1305 ▪ See IETF RFC 5246 clause 6.3
  - 1306 ▪ The length of key\_block depends on cipher suite.
    - 1307 • (e.g. 96 bytes for TLS\_ECDHE\_ECDSA\_WITH\_AES\_128\_CBC\_SHA256
    - 1308 40 bytes for TLS\_PSK\_WITH\_AES\_128\_CCM\_8)
- 1309 - Message is a concatenation of the following:
  - 1310 ▪ DoxmType string for the current onboarding method (e.g. "oic.sec.doxm.jw")
    - 1311 • See clause 13.2.2 for specific DoxmTypes
  - 1312 ▪ Owner ID is a UUID identifying the device owner identifier and the device that maintains SharedKey.
    - 1313 • Use raw bytes as specified in IETF RFC 4122 clause 4.1.2
  - 1314 ▪ Device ID is new device's UUID Device ID
    - 1315 • Use raw bytes as specified in IETF RFC 4122 clause 4.1.2
- 1316 - SharedKey Length will be 32 octets.
  - 1317 ▪ If subsequent DTLS sessions use 128 bit encryption cipher suites the left most 16 octets will be used.
  - 1318 ▪ DTLS sessions using 256-bit encryption cipher suites will use all 32 octets.

1319 **7.3.3 Certificate Credential Generation**

1320 The Certificate Credential will be used by Devices for secure bidirectional communication. The  
1321 certificates will be issued by a CMS or an external certificate authority (CA). This CA will be used  
1322 to mutually establish the authenticity of the Device.

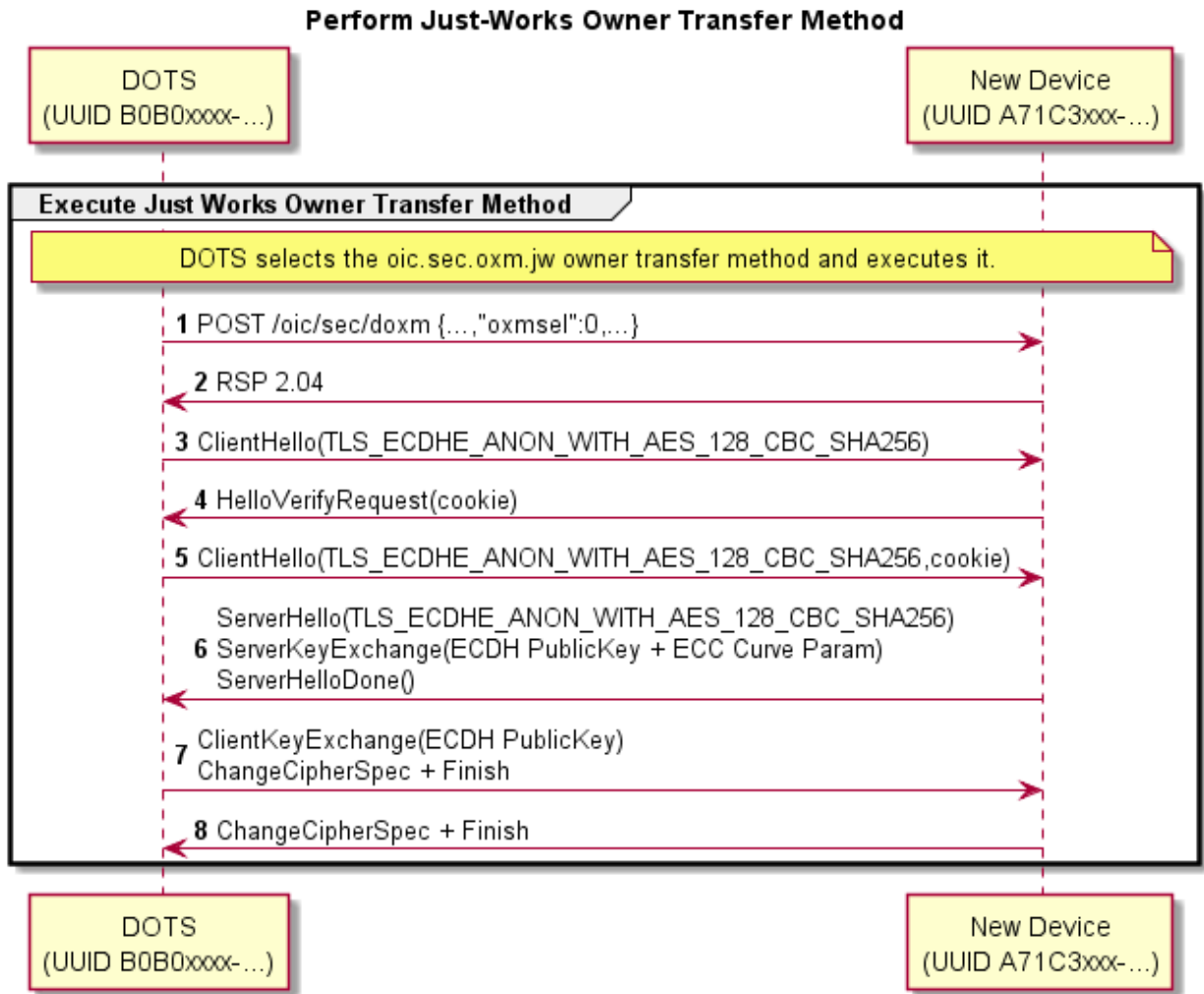
1323 **7.3.4 Just-Works OTM**

1324 **7.3.4.1 Just-Works OTM General**

1325 Just-works OTM creates a symmetric key credential that is a pre-shared key used to establish a  
1326 secure connection through which a device should be provisioned for use within the owner's OCF  
1327 Security Domain. Provisioning additional credentials and Resources is a typical step following  
1328 ownership establishment. The pre-shared key is called SharedKey.

1329 The DOTS selects the Just-works OTM using the "oxmsel" Property of the "/oic/sec/doxm"  
1330 Resource and establishes a DTLS session using a ciphersuite defined for the Just-works OTM.

1331 Just Works OTM sequence is shown in Figure 12 and steps described in Table 2.



1332

1333

1334

**Figure 12 – A Just Works OTM**

1335

**Table 2 – A Just Works OTM Details**

Step	Description
1, 2	The DOTS notifies the Device that it selected the "Just Works" method.
3 - 8	A DTLS session is established using anonymous Diffie-Hellman. <sup>a</sup>
<sup>a</sup> This method assumes the operator is aware of the potential for man-in-the-middle attack and has taken precautions to perform the method in a clean-room network.	

1336 **7.3.4.2 Security Considerations**

1337 Anonymous Diffie-Hellman key agreement is subject to a man-in-the-middle attacker. Use of this  
1338 method presumes that both the DOTS and the new device perform the "just-works" method  
1339 assumes onboarding happens in a relatively safe environment absent of an attack device.

1340 This method doesn't have a trustworthy way to prove the device ID asserted is reliably bound to  
1341 the device.

1342 The new device should use a temporal device ID prior to transitioning to an owned device while it  
1343 is considered a guest device to prevent privacy sensitive tracking. The device asserts a non-  
1344 temporal device ID that could differ from the temporal value during the secure session in which  
1345 owner transfer exchange takes place. The DOTS verifies the asserted Device ID does not conflict  
1346 with a Device ID already in use. If it is already in use the existing credentials are used to establish  
1347 a secure session.

1348 An un-owned Device that also has established device credentials might be an indication of a  
1349 corrupted or compromised device.

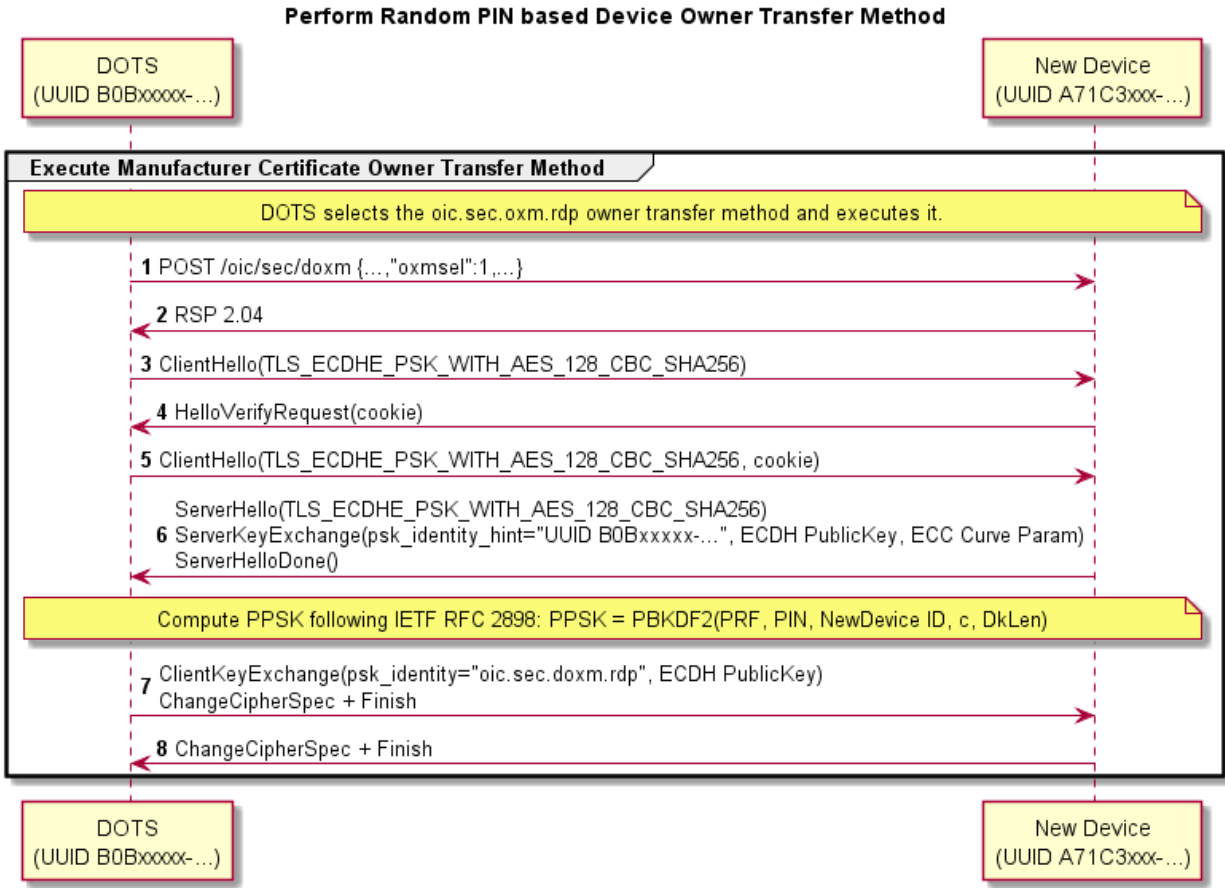
1350 **7.3.5 Random PIN based OTM**

1351 **7.3.5.1 Random PIN based OTM General**

1352 The Random PIN method establishes physical proximity between the new device and the OBT can  
1353 prevent man-in-the-middle attacks. The Device generates a random number that is communicated  
1354 to the DOTS over an Out of Band Communication Channel. The definition of an Out of Band  
1355 Communication Channel is outside the scope of the definition of device OTMs. The DOTS and new  
1356 Device use the PIN in a key exchange as evidence that someone authorized the transfer of  
1357 ownership by having physical access to the new Device via the Out-of-Band Communication  
1358 Channel.

1359 **7.3.5.2 Random PIN based Owner Transfer Sequence**

1360 Random PIN-based OTM sequence is shown in Figure 13 and steps described in Table 3.



1361  
1362  
1363  
1364

**Figure 13 – Random PIN-based OTM**

**Table 3 – Random PIN-based OTM Details**

Step	Description
1, 2	The DOTS notifies the Device that it selected the "Random PIN" method.
3 - 8	A DTLS session is established using PSK-based Diffie-Hellman ciphersuite. The PIN is supplied as the PSK parameter. The PIN is randomly generated by the new device then communicated via an Out of Band Communication Channel that establishes proximal context between the new device and the DOTS. The security principle is the attack device will be unable to intercept the PIN due to a lack of proximity.

- 1365 The following requirements apply to the DTLS handshake messages for this OTM:
- 1366 – The new Device shall set the "psk\_identity\_hint" field of the ServerKeyExchange message to
  - 1367 the "deviceuuid" Property of the "/oic/sec/doxm" Resource being sent in responses when the
  - 1368 new Device is in RFOTM and when a Device Onboarding Connection is not currently
  - 1369 established.
  - 1370 – The new Device determines that the Random PIN-based OTM is being applied if that the
  - 1371 "psk\_identity" field of the ClientKeyExchange message matches the string "oic.sec.doxm.rdp".

1372 If the Random PIN-based OTM is being applied, then the new Device shall apply the key  
1373 derivation below.

1374 NOTE The string "oic.sec.doxm.rdp" is the URN defined for the Random PIN-based OTM in Table 18 and is included to  
1375 allow future OTMs to re-use the DTLS ciphersuites without confusion about which OTM should be applied.

1376 This OTM uses a pseudo-random function (PBKDF2) defined by IETF RFC 2898 and a PIN  
1377 exchanged via an Out of Band Communication Channel to generate a pre-shared key. The PIN-  
1378 authenticated pre-shared key (PPSK) is supplied to TLS ciphersuites that accept a PSK.

1379 – PPSK = PBKDF2(PRF, PIN, Device ID, c, dkLen)

1380 The PBKDF2 function has the following parameters:

1381 – PRF – Uses the TLS 1.2 PRF defined by IETF RFC 5246.

1382 – PIN – obtained via Out of Band Communication Channel.

1383 – Device ID – the "deviceuuid" Property of the "/oic/sec/doxm" Resource being sent in responses  
1384 when the new Device is in RFOTM and when a Device Onboarding Connection is not currently  
1385 established.

1386 Use raw bytes as specified in IETF RFC 4122 clause 4.1.2

1387 – c – Iteration count initialized to 1000

1388 – dkLen – Desired length of the derived PSK in octets.

### 1389 **7.3.5.3 Security Considerations**

1390 Security of the Random PIN mechanism depends on the entropy of the PIN. Using a PIN with  
1391 insufficient entropy may allow a man-in-the-middle attack to recover any long-term credentials  
1392 provisioned as a part of onboarding. In particular, learning the provisioned symmetric key  
1393 credentials allows an attacker to masquerade as the onboarded device.

1394 It is recommended that the entropy of the PIN be enough to withstand an online brute-force attack,  
1395 40 bits or more. For example, a 12-digit numeric PIN, or an 8-character alphanumeric (0-9a-z), or  
1396 a 7-character case-sensitive alphanumeric PIN (0-9a-zA-Z). A man-in-the-middle attack (MITM) is  
1397 when the attacker is active on the network and can intercept and modify messages between the  
1398 DOTS and device. In the MITM attack, the attacker must recover the PIN from the key exchange  
1399 messages in "real time", i.e., before the peer's time out and abort the connection attempt. Having  
1400 recovered the PIN, he can complete the authentication step of key exchange. The guidance given  
1401 here calls for a minimum of 40 bits of entropy, however, the assurance this provides depends on  
1402 the resources available to the attacker. Given the parallelizable nature of a brute force guessing  
1403 attack, the attack enjoys a linear speedup as more cores/threads are added. A more conservative  
1404 amount of entropy would be 64 bits. Since the Random PIN OTM requires using a DTLS ciphersuite  
1405 that includes an ECDHE key exchange, the security of the Random PIN OTM is always at least  
1406 equivalent to the security of the JustWorks OTM.

1407 The Random PIN OTM also has an option to use PBKDF2 to derive key material from the PIN. The  
1408 rationale is to increase the cost of a brute force attack, by increasing the cost of each guess in the  
1409 attack by a tuneable amount (the number of PBKDF2 iterations). In theory, this is an effective way  
1410 to reduce the entropy requirement of the PIN. Unfortunately, it is difficult to quantify the reduction,  
1411 since an X-fold increase in time spent by the honest peers does not directly translate to an X-fold  
1412 increase in time by the attacker. This asymmetry is because the attacker may use specialized  
1413 implementations and hardware not available to honest peers. For this reason, when deciding how  
1414 much entropy to use for a PIN, it is recommended that implementers assume PBKDF2 provides no  
1415 security, and ensure the PIN has sufficient entropy.

1416 The Random PIN device OTM security depends on an assumption that a secure Out of Band  
1417 Communication Channel for communicating a randomly generated PIN from the new device to the  
1418 OBT exists. If the Out of Band Communication Channel leaks some or the entire PIN to an attacker,

1419 this reduces the entropy of the PIN, and the attacks described above apply. The Out of Band  
1420 Communication Channel should be chosen such that it requires proximity between the DOTS and  
1421 the new device. The attacker is assumed to not have compromised the Out of Band Communication  
1422 Channel. As an example Out of Band Communication Channel, the device may display a PIN to be  
1423 entered into the OBT software. Another example is for the device to encode the PIN as a 2D  
1424 barcode and display it for a camera on the DOTS device to capture and decode.

### 1425 **7.3.6 Manufacturer Certificate Based OTM**

#### 1426 **7.3.6.1 Manufacturer Certificate Based OTM General**

1427 The manufacturer certificate-based OTM shall use a certificate embedded into the device by the  
1428 manufacturer and may use a signed OBT, which determines the Trust Anchor between the device  
1429 and the DOTS.

1430 Manufacturer embedded certificates do not necessarily need to chain to an OCF Root CA trust  
1431 anchor.

1432 For some environments, policies or administrators, additional information about device  
1433 characteristics may be sought. This list of additional attestations that OCF may or may not have  
1434 tested (understanding that some attestations are incapable of testing or for which testing may be  
1435 infeasible or economically unviable) can be found under the OCF Security Claims x509.v3  
1436 extension described in 9.4.2.2.6.

1437 When utilizing certificate-based ownership transfer, devices shall utilize asymmetric keys with  
1438 certificate data to authenticate their identities with the DOTS in the process of bringing a new  
1439 device into operation on an OCF Security Domain. The onboarding process involves several  
1440 discrete steps:

#### 1441 1) Pre-on-board conditions

1442 a) The credential element of the Device's credential Resource ("/oic/sec/cred") containing the  
1443 manufacturer certificate shall be identified by the "credusage" Property containing the string  
1444 "oic.sec.cred.mfgcert" to indicate that the credential contains a manufacturer certificate.

1445 b) The manufacturer certificate chain shall be contained in the identified credential element's  
1446 "publicdata" Property.

1447 c) The device shall contain a unique and immutable ECC asymmetric key pair.

1448 d) If the device requires authentication of the DOTS as part of ownership transfer, it is  
1449 presumed that the DOTS has been registered and has obtained a certificate for its unique  
1450 and immutable ECC asymmetric key pair signed by the predetermined Trust Anchor.

1451 e) User has configured the DOTS app with network access info and account info (if any).

1452 2) The DOTS authenticates the Device using ECDSA to verify the signature. Additionally, the  
1453 Device may authenticate the DOTS to verify the DOTS signature.

1454 3) If authentication fails, the Device shall indicate the reason for failure and return to the Ready  
1455 for OTM state. If authentication succeeds, the Device shall establish an encrypted link with the  
1456 DOTS in accordance with the negotiated cipher suite.

#### 1457 **7.3.6.2 Certificate Profiles**

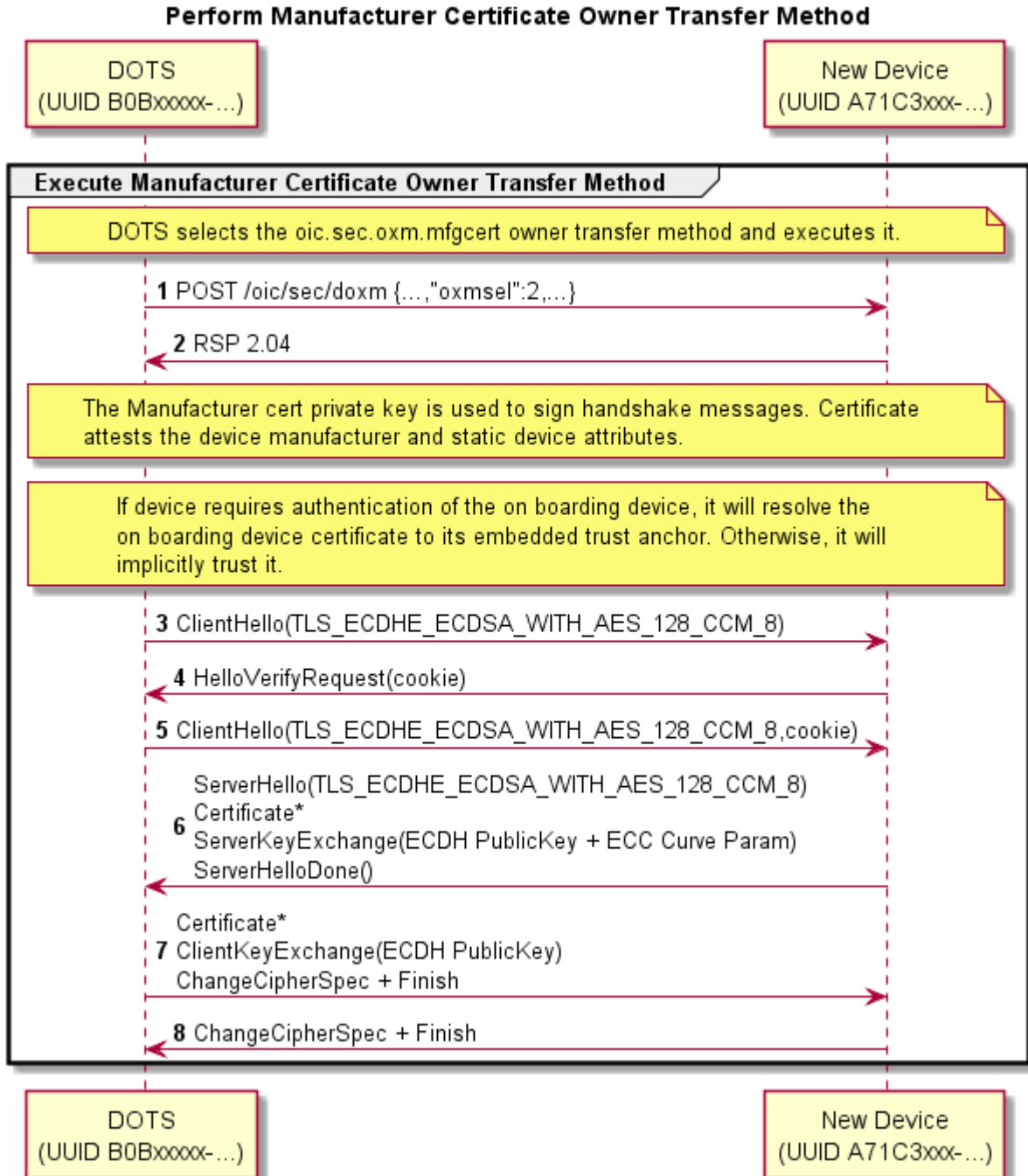
1458 See 9.4.2 for details.

#### 1459 **7.3.6.3 Certificate Owner Transfer Sequence Security Considerations**

1460  
1461 The OBT shall authenticate the device during onboarding. The device will not authenticate the OBT.  
1462 During the DTLS handshake the server shall not send a Certificate Request.

1463 **7.3.6.4 Manufacturer Certificate Based OTM Sequence**

1464 Manufacturer Certificate Based OTM sequence is shown in Figure 14 and steps described in  
 1465 Table 4.



1466

1467

**Figure 14 – Manufacturer Certificate Based OTM Sequence**

1468



**Table 4 – Manufacturer Certificate Based OTM Details**

Step	Description
1, 2	The DOTS notifies the Device that it selected the "Manufacturer Certificate" method.
3 - 8	A DTLS session is established using the device's manufacturer certificate and optional DOTS certificate. The device's manufacturer certificate may contain data attesting to the Device hardening and security properties.

1470 **7.3.6.5 Security Considerations**

1471 The manufacturer certificate private key is embedded in the Platform with a sufficient degree of  
1472 assurance that the private key cannot be compromised.

1473 The Platform manufacturer issues the manufacturer certificate and attests the private key  
1474 protection mechanism.

1475 **7.3.7 Vendor Specific OTMs**

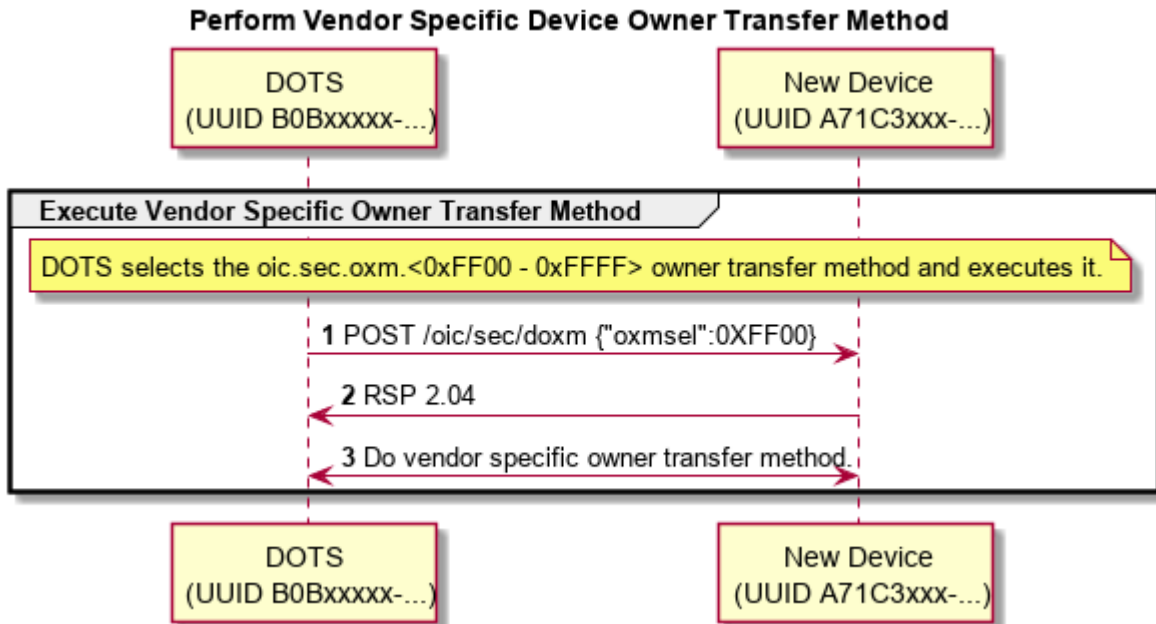
1476 **7.3.7.1 Vendor Specific OTM General**

1477 The OCF anticipates situations where a vendor will need to implement an OTM that accommodates  
1478 manufacturing or Device constraints. The Device OTM resource is extensible for this purpose.  
1479 Vendor-specific OTMs must adhere to a set of conventions that all OTMs follow.

- 1480 – The OBT must determine which credential types are supported by the Device. This is  
1481 accomplished by querying the Device's "/oic/sec/doxm" Resource to identify supported  
1482 credential types.
- 1483 – The OBT provisions the Device with OC(s).
- 1484 – The OBT supplies the Device ID and credentials for subsequent access to the OBT.
- 1485 – The OBT will supply second carrier settings sufficient for accessing the owner's OCF Security  
1486 Domain subsequent to ownership establishment.
- 1487 – The OBT may perform additional provisioning steps but must not invalidate provisioning tasks  
1488 to be performed by a security service.

1489 **7.3.7.2 Vendor-specific Owner Transfer Sequence Example**

1490 Vendor-specific OTM sequence example is shown in Figure 15 and steps described in Table 5.



**Figure 15 – Vendor-specific Owner Transfer Sequence**

**Table 5 – Vendor-specific Owner Transfer Details**

Step	Description
1, 2	The DOTS selects a vendor-specific OTM.
3	The vendor-specific OTM is applied

**7.3.7.3 Security Considerations**

The vendor is responsible for considering security threats and mitigation strategies.

**7.3.8 Establishing Owner Credentials**

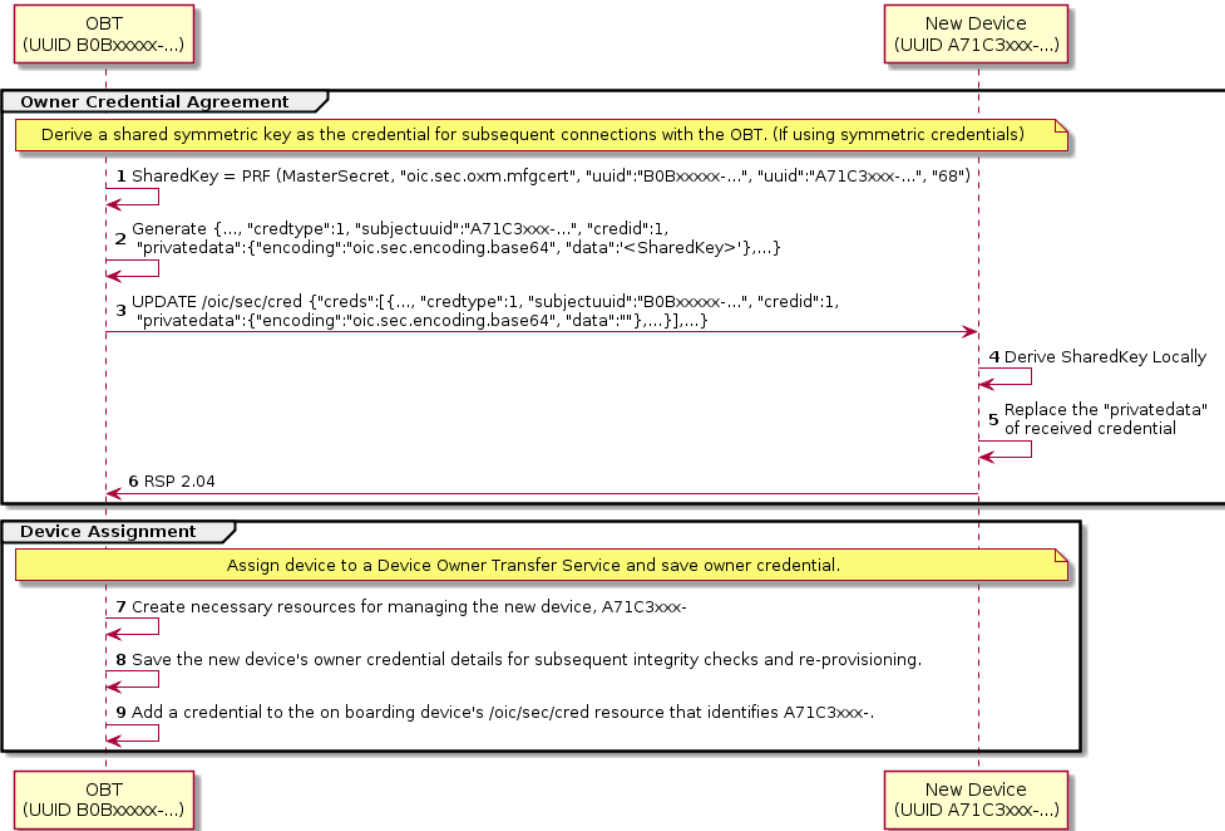
Once the OBT and the new Device have authenticated and established an encrypted connection using one of the defined OTM methods, the Owner Credential(s) can be provisioned.

The Owner Credential is provisioned as part of Ownership Transfer Method, and may be provisioned directly by CMS.

The steps for establishing Device's owner credentials (OC) as part of OTM are:

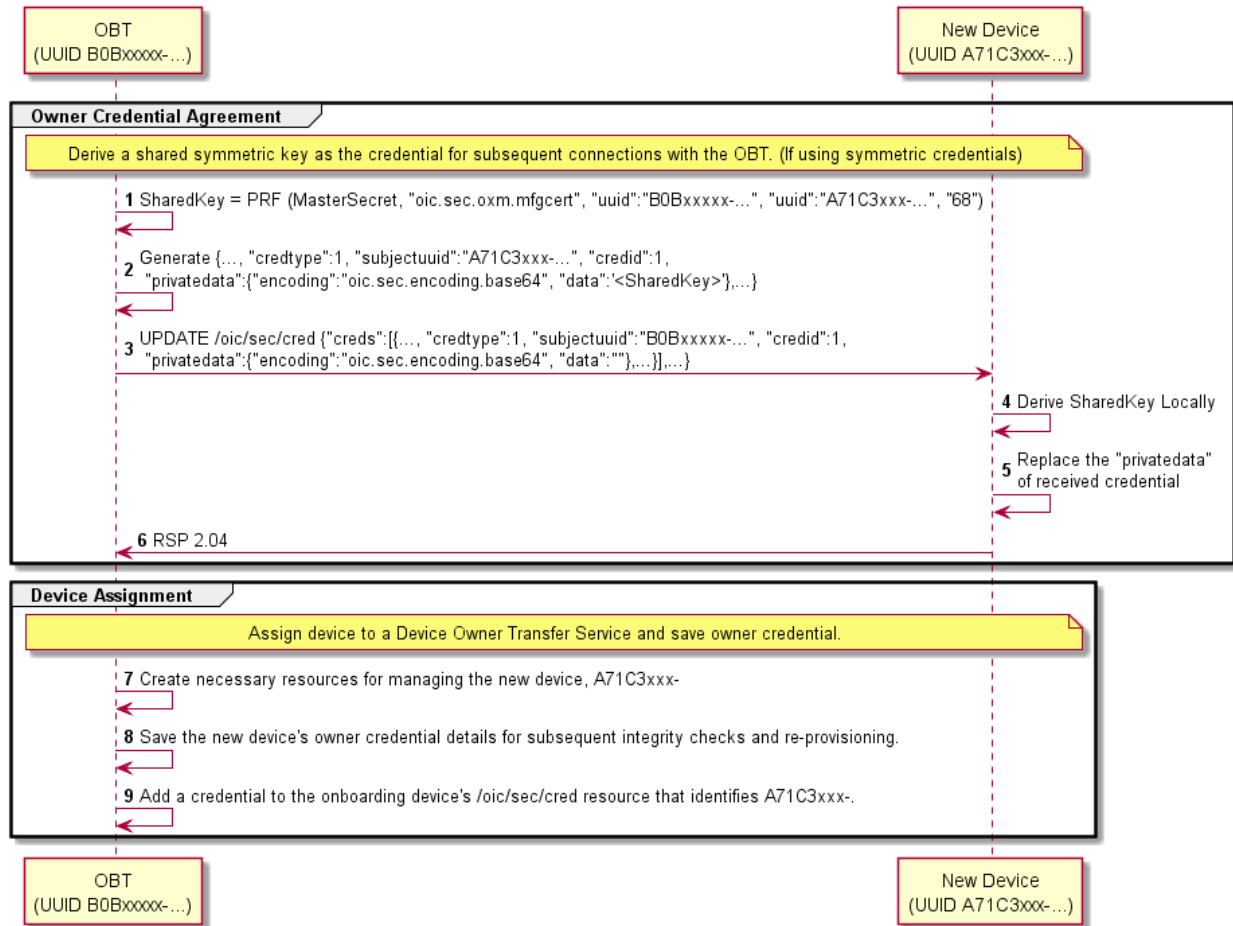
- 1) The OBT establishes the Device ID and Device Owner Id.
- 2) The OBT then establishes Device's symmetric OC - See Figure 16 and Table 6.
- 3) Configure Device services.
- 4) Configure Device for peer to peer interaction.

### Symmetric Owner Credential (OC) Assignment Sequence



1508

### Symmetric Owner Credential (OC) Assignment Sequence



1509

1510

1511

1512

**Figure 16 – Symmetric Owner Credential Provisioning Sequence**

**Table 6 – Symmetric Owner Credential Assignment Details**

Step	Description
1, 2	The OBT uses a pseudo-random-function (PRF), the master secret resulting from the DTLS handshake, and other information to generate a symmetric key credential resource Property - SharedKey.
3	The OBT creates a credential resource Property set based on SharedKey and then sends the resource Property set to the new Device with empty "privatedata" Property value.
4, 5	The new Device locally generates the SharedKey and updates it to the "privatedata" Property of the credential resource Property set.
6	The new Device sends a success message.
7	The onboarding service creates a subjects resource for the new device (e.g./A71C3xxx-...)
8	The onboarding service provisions its "/oic/svc/dots/subjects/A71C3xxx-/cred" resource with

	the owner credential. Credential type is SYMMETRIC KEY.
9	(optional) The onboarding service provisions its own "/oic/sec/cred" resource with the owner credential for new device. Credential type is SYMMETRIC KEY.

- 1513 In particular when OBT establishes symmetric owner credentials as part of OTM sequence:
- 1514 – The OBT generates a Shared Key using the SharedKey Credential Calculation method
  - 1515 described in 7.3.2.
  - 1516 – The OBT sends an empty key to the new Device's "/oic/sec/cred" Resource, identified as a
  - 1517 symmetric pair-wise key. The Subject UUID of the "/oic/sec/cred" entry shall match the Device
  - 1518 UUID of the OBT.
  - 1519 – Upon receipt of the OBT's symmetric owner credential, the new Device shall independently
  - 1520 generate the Shared Key using the SharedKey Credential Calculation method described in 7.3.2
  - 1521 and store it with the owner credential.
  - 1522 – The new Device shall use the Shared Key owner credential(s) stored via the "/oic/sec/cred"
  - 1523 Resource to authenticate the owner during subsequent connections.

1524 **7.3.9 Security considerations regarding selecting an Ownership Transfer Method -**

1525 **Moved to OCF Onboarding Tool document**

1526 This clause is intentionally left blank.

1527 **7.3.10 Security Profile Assignment**

1528 OCF Devices may have been evaluated according to an OCF Security Profile. Evaluation results

1529 could be accessed from a manufacturer's certificate, OCF web server or other public repository.

1530 The DOTS reviews evaluation results to determine which OCF Security Profiles the OCF Device is

1531 authorized to possess and configures the Device with the subset of evaluated security profiles best

1532 suited for the OCF Security Domain owner's intended segmentation strategy.

1533 The OCF Device vendor shall set a manufacturer default value for the "supportedprofiles" Property

1534 of the "/oic/sec/sp" Resource to match those approved by OCF's testing and certification process.

1535 The "currentprofile" Property of the "/oic/sec/sp" Resource shall be set to one of the values

1536 contained in the "supportedprofiles". The manufacturer default value shall be re-asserted when the

1537 Device transitions to RESET Device State.

1538 The OCF Device shall only allow the "/oic/sec/sp" Resource to be updated when the Device is in

1539 one of the following Device States: RFOTM, RFPRO, SRESET and may not allow any update as

1540 directed by a Security Profile.

1541 The DOTS may update the "supportedprofiles" Property of the "/oic/sec/sp" Resource with a subset

1542 of the OCF Security Profiles values the Device achieved as part of OCF Conformance testing. The

1543 DOTS may locate conformance results by inspecting manufacturer certificates supplied with the

1544 OCF Device by selecting the "credusage" Property of the "/oic/sec/cred" Resource having the value

1545 of "oic.sec.cred.mfgcert". The DOTS may further locate conformance results by visiting a well-

1546 known OCF web site URI corresponding to the ocfCPLAttributes extension fields (clause 9.4.2.2.7).

1547 The DOTS may select a subset of Security Profiles (from those evaluated by OCF conformance

1548 testing) based on a local policy.

1549 As part of onboarding (while the OTM session is active) the DOTS should configure ACE entries to

1550 allow DOTS access subsequent to onboarding.

1551 The DOTS should update the "currentprofile" Property of the "/oic/sec/sp" Resource with the value

1552 that most correctly depicts the OCF Security Domain owner's intended Device deployment strategy.

1553 The CMS may issue role credentials using the Security Profile value (e.g. the "sp-blue-v0 OID") to  
1554 indicate the OCF Security Domain owner's intention to segment the OCF Security Domain  
1555 according to a Security Profile. The CMS retrieves the supportedprofiles Property of the  
1556 "/oic/sec/sp" Resource to select role names corroborated with the Device's supported Security  
1557 Profiles when issuing role credentials.

1558 If the CMS issues role credentials based on a Security Profile, the AMS supplies access control  
1559 entries that include the role designation(s).

## 1560 **7.4 Provisioning**

### 1561 **7.4.1 Provisioning Flows**

#### 1562 **7.4.1.1 Provisioning Flows General**

1563 As part of onboarding a new Device a secure channel is formed between the new Device and the  
1564 OBT. Subsequent to the Device ownership status being changed to "owned", there is an opportunity  
1565 to begin provisioning. The OBT provisions the support services that should be subsequently used  
1566 to complete Device provisioning and on-going Device management.

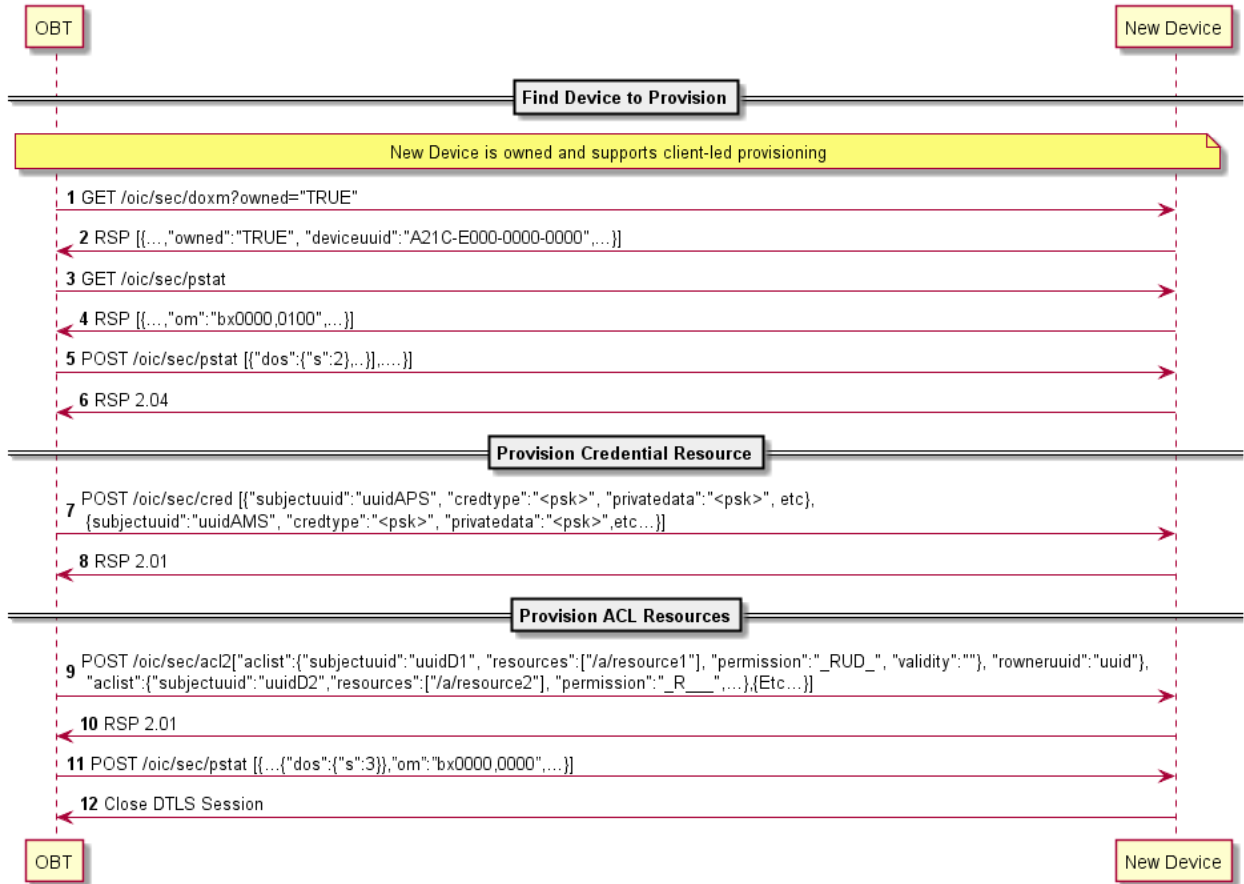
1567 The Device employs a Client-directed provisioning strategy. The "/oic/sec/pstat" Resource  
1568 identifies the provisioning strategy and current provisioning status. The provisioning service should  
1569 determine which provisioning strategy is most appropriate for the OCF Security Domain. See 13.8  
1570 for additional detail.

#### 1571 **7.4.1.2 Client-directed Provisioning**

1572 Client-directed provisioning relies on a provisioning service that identifies Servers in need of  
1573 provisioning then performs all necessary provisioning duties.

1574 An example of Client-directed provisioning is shown in Figure 17 and steps described in Table 7.

**OCF Client-directed Provisioning  
with a Single Service Provider**



1575

1576

1577

1578

**Figure 17 – Example of Client-directed provisioning**

**Table 7 – Steps describing Client -directed provisioning**

Step	Description
1	Discover Devices that are owned and support Client-directed provisioning.
2	The "/oic/sec/doxm" Resource identifies the Device and it's owned status.
3	DOTS (on OBT) obtains the new Device's provisioning status found in "/oic/sec/pstat" Resource
4	The "pstat" Resource describes the types of provisioning modes supported and which is currently configured. A Device manufacturer should set a default current operational mode ("om"). If the "om" isn't configured for Client-directed provisioning, its "om" value can be changed.
5 - 6	Change Device state to Ready-for-Provisioning.
7 - 8	CMS (on OBT)instantiates the "/oic/sec/cred" Resource. It contains credentials for the provisioned services and other Devices

9 - 10	AMS (on OBT) instantiates "/oic/sec/acl2" Resource.
11	The new Device provisioning status mode is updated to reflect that ACLs have been configured. (Ready-for-Normal-Operation state)
12	The secure session is closed.

1579 **7.4.1.3 Server-directed Provisioning [DEPRECATED]**

1580 This clause is intentionally left blank.

1581 **7.4.1.4 Server-directed Provisioning Involving Multiple Support Services [DEPRECATED]**

1582 This clause is intentionally left blank.

1583 **7.5 Device Provisioning for OCF Cloud – moved to OCF Cloud Security document**

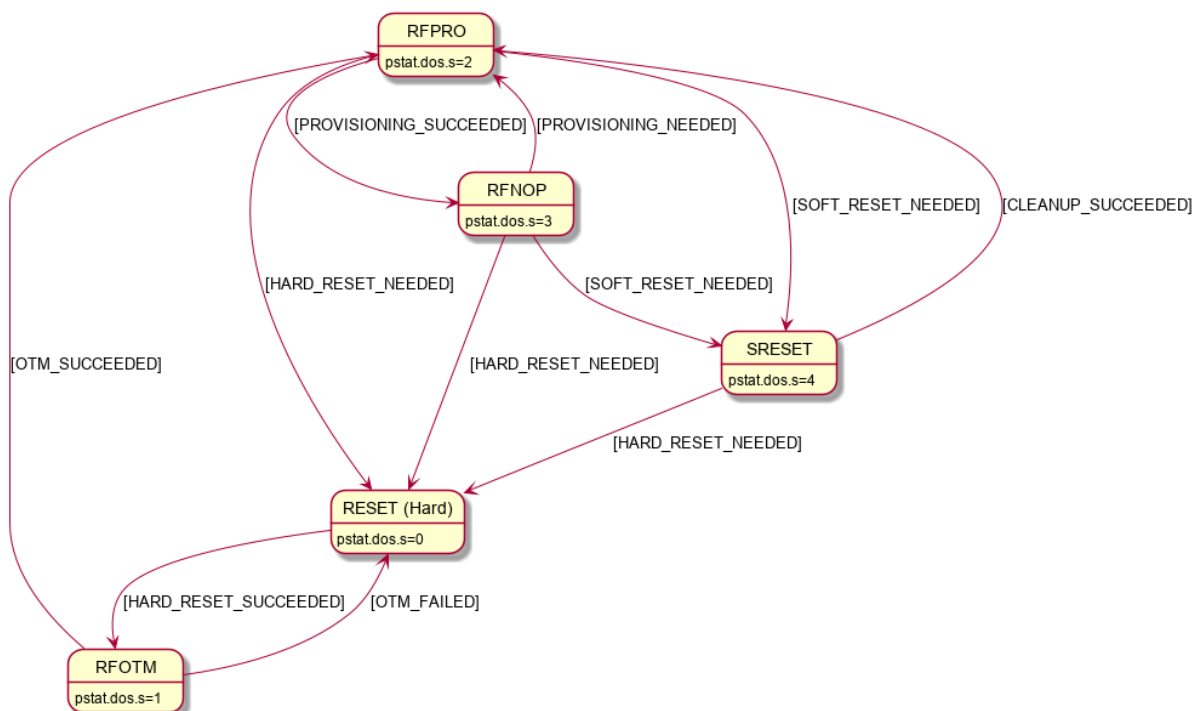
1584 This clause is intentionally left blank.

1585 **8 Device Onboarding State Definitions**

1586 **8.1 Device Onboarding General**

1587 As explained in 5.3, the process of onboarding completes after the ownership of the Device has been transferred and the Device has been provisioned with relevant configuration/services as explained in 5.4. The Figure 18 shows the various states a Device can be in during the Device lifecycle. Device shall reject any requests to perform a state transition not shown on Figure 18.

1592 The "/pstat.dos.s" Property is RW by the "/oic/sec/pstat" resource owner (e.g. "doxs" service) so that the resource owner can remotely update the Device state. When the Device is in RFNOP or RFPRO, ACLs can be used to allow remote control of Device state by other Devices. When the Device state is SRESET the Device OC may be the only indication of authorization to access the Device. The Device owner may perform low-level consistency checks and re-provisioning to get the Device suitable for a transition to RFPRO.



1598



1599

## Figure 18 – Device state model

1600 As shown in the diagram, at the conclusion of the provisioning step, the Device comes in the "Ready  
1601 for Normal Operation" state where it has all it needs in order to start interoperating with other  
1602 Devices. Clause 8.5 specifies the minimum mandatory configuration that a Device shall hold in  
1603 order to be considered as "Ready for Normal Operation".

1604 In the event of power loss or Device failure, the Device should remain in the same state that it was  
1605 in prior to the power loss / failure

1606 If a Device or resource owner OBSERVEs "/pstat.dos.s", then transitions to SRESET will give early  
1607 warning notification of Devices that may require SVR consistency checking.

1608 In order for onboarding to function, the Device shall have the following Resources installed:

- 1609 1) "/oic/sec/doxm" Resource
- 1610 2) "/oic/sec/pstat" Resource
- 1611 3) "/oic/sec/cred" Resource

1612 The values contained in these Resources are specified in the state definitions in 8.2, 8.3, 8.4, 8.5  
1613 and 8.6.

### 1614 8.2 Device Onboarding-Reset State Definition

1615 The /pstat.dos.s = RESET state is defined as a "hard" reset to manufacturer defaults. Hard reset  
1616 also defines a state where the Device asset is ready to be transferred to another party.

1617 The Platform manufacturer should provide a physical mechanism (e.g. button) that forces Platform  
1618 reset. All Devices hosted on the same Platform transition their Device states to RESET when the  
1619 Platform reset is asserted.

1620 The following Resources and their specific properties shall have the value as specified:

- 1621 – The "owned" Property of the "/oic/sec/doxm" Resource shall transition to FALSE.
- 1622 – The "devowneruuid" Property of the "/oic/sec/doxm" Resource shall be nil UUID.
- 1623 – The "deviceuuid" Property of the "/oic/sec/doxm" Resource shall be set to the manufacturer  
1624 default value.
- 1625 – The "sct" Property of the "/oic/sec/doxm" Resource shall be reset to the manufacturer's default  
1626 value.
- 1627 – The "oxmsel" Property of the "/oic/sec/doxm" Resource shall be reset to the manufacturer's  
1628 default value.
- 1629 – The "isop" Property of the "/oic/sec/pstat" Resource shall be FALSE.
- 1630 – The "dos" Property of the "/oic/sec/pstat" Resource shall be updated: dos.s shall equal "RESET"  
1631 state.
- 1632 – The "om" (operational modes) Property of the "/oic/sec/pstat" Resource shall be set to the  
1633 manufacturer default value.
- 1634 – The "sm" (supported operational modes) Property of the "/oic/sec/pstat" Resource shall be set  
1635 to the manufacturer default value.
- 1636 – The "rowneruuid" Property of "/oic/sec/pstat", "/oic/sec/doxm", "/oic/sec/acl2", and  
1637 "/oic/sec/cred" Resources shall be nil UUID.
- 1638 – The "supportedprofiles" Property of the "/oic/sec/sp" Resource shall be set to the manufacturer  
1639 default value.

- 1640 – The "currentprofile" Property of the "/oic/sec/sp" Resource shall be set to the manufacturer  
1641 default value.

### 1642 **8.3 Device Ready-for-OTM State Definition**

1643 The following Resources and their specific properties shall have the value as specified when the  
1644 Device enters ready for ownership transfer:

- 1645 – The "owned" Property of the "/oic/sec/doxm" Resource shall be FALSE and will transition to  
1646 TRUE.
- 1647 – The "devowneruuid" Property of the "/oic/sec/doxm" Resource shall be nil UUID.
- 1648 – The "deviceuuid" Property of the "/oic/sec/doxm" Resource shall be set to the manufacturer  
1649 default value.
- 1650 – The "isop" Property of the "/oic/sec/pstat" Resource shall be FALSE.
- 1651 – The "dos" of the "/oic/sec/pstat" Resource shall be updated: "dos.s" shall equal "RFOTM" state.
- 1652 – The "/oic/sec/cred" Resource shall contain credential(s) if required by the selected OTM

### 1653 **8.4 Device Ready-for-Provisioning State Definition**

1654 The following Resources and their specific properties shall have the value as specified when the  
1655 Device enters ready for provisioning:

- 1656 – The "owned" Property of the "/oic/sec/doxm" Resource shall be TRUE.
- 1657 – The "devowneruuid" Property of the "/oic/sec/doxm" Resource shall not be nil UUID.
- 1658 – The "deviceuuid" Property of the "/oic/sec/doxm" Resource shall not be nil UUID and shall be  
1659 set to the value that was determined during RFOTM processing.
- 1660 – The "oxmsel" Property of the "/oic/sec/doxm" Resource shall have the value of the actual OTM  
1661 used during ownership transfer.
- 1662 – The "isop" Property of the "/oic/sec/pstat" Resource shall be FALSE.
- 1663 – The "dos" of the "/oic/sec/pstat" Resource shall be updated: "dos.s" shall equal "RFPRO" state.
- 1664 – The "rowneruuid" Property of every installed Resource shall be set to a valid Resource owner  
1665 (i.e. an entity that is authorized to instantiate or update the given Resource). Failure to set a  
1666 "rowneruuid" may result in an orphan Resource.
- 1667 – The "/oic/sec/cred" Resource shall contain credentials for each entity referenced by  
1668 "rowneruuid" and "devowneruuid" Properties.
- 1669 – All requests to the "/oic/sec/roles" Resource received over a mutually-authenticated connection  
1670 established using an identity certificate shall be granted, regardless of the configuration of the  
1671 ACEs in the "/oic/sec/acl2" Resource, subject to the conditions in clause 10.4.2.

### 1672 **8.5 Device Ready-for-Normal-Operation State Definition**

1673 The following Resources and their specific properties shall have the value as specified when the  
1674 Device enters ready for normal operation:

- 1675 – The "owned" Property of the "/oic/sec/doxm" Resource shall be TRUE.
- 1676 – The "devowneruuid" Property of the "/oic/sec/doxm" Resource shall not be nil UUID.
- 1677 – The "deviceuuid" Property of the "/oic/sec/doxm" Resource shall not be nil UUID and shall be  
1678 set to the ID that was configured during OTM. Also the value of the "di" Property in "/oic/d" shall  
1679 be the same as the deviceuuid.
- 1680 – The "oxmsel" Property of the "/oic/sec/doxm" Resource shall have the value of the actual OTM  
1681 used during ownership transfer.

- 1682 – The "isop" Property of the "/oic/sec/pstat" Resource shall be set to TRUE by the Server once  
1683 transition to RFNOP is otherwise complete.
- 1684 – The "dos" of the "/oic/sec/pstat" Resource shall be updated: "dos.s" shall equal "RFNOP" state.
- 1685 – The "rowneruuid" Property of every installed Resource shall be set to a valid resource owner  
1686 (i.e. an entity that is authorized to instantiate or update the given Resource). Failure to set a  
1687 "rowneruuid" results in an orphan Resource.
- 1688 – The "/oic/sec/cred" Resource shall contain credentials for each service referenced by  
1689 "rowneruuid" and "devowneruuid" Properties.
- 1690 – All requests to the "/oic/sec/roles" Resource received over a mutually-authenticated connection  
1691 established using an identity certificate shall be granted, regardless of the configuration of the  
1692 ACEs in the "/oic/sec/acl2" Resource, subject to the conditions in clause 10.4.2.

### 1693 **8.6 Device Soft Reset State Definition**

1694 The soft reset state is defined (e.g. "/pstat.dos.s" = SRESET) where entrance into this state means  
1695 the Device is not operational but remains owned by the current owner. The Device may exit  
1696 SRESET by authenticating to a DOTS (e.g. "rt" = "oic.r.doxs") using the OC provided during original  
1697 onboarding (but should not require use of an OTM /doxm.oxms).

1698 If the DOTS credential cannot be found or is determined to be corrupted, the Device state  
1699 transitions to RESET. The Device should remain in SRESET if the DOTS credential fails to validate  
1700 the DOTS. This mitigates denial-of-service attacks that may be attempted by non-DOTS Devices.

1701 When in SRESET, the following Resources and their specific Properties shall have the values as  
1702 specified.

- 1703 – The "owned" Property of the "/oic/sec/doxm" Resource shall be TRUE.
- 1704 – The "devowneruuid" Property of the "/oic/sec/doxm" Resource shall remain non-null.
- 1705 – The "deviceuuid" Property of the "/oic/sec/doxm" Resource shall remain non-null.
- 1706 – The "sct" Property of the "/oic/sec/doxm" Resource shall retain its value.
- 1707 – The "oxmsel" Property of the "/oic/sec/doxm" Resource shall retains its value.
- 1708 – The "isop" Property of the "/oic/sec/pstat" Resource shall be FALSE.
- 1709 – The "/oic/sec/pstat.dos.s" Property shall be SRESET.
- 1710 – The "om" (operational modes) Property of the "/oic/sec/pstat" Resource shall be "client-directed  
1711 mode".
- 1712 – The "sm" (supported operational modes) Property of "/oic/sec/pstat" Resource may be updated  
1713 by the Device owner (aka DOTS).
- 1714 – The "rowneruuid" Property of "/oic/sec/pstat", "/oic/sec/doxm", "/oic/sec/acl2", and  
1715 "/oic/sec/cred" Resources may be reset by the Device owner (aka DOTS) and re-provisioned.
- 1716 – All requests to the "/oic/sec/roles" Resource received over a mutually-authenticated connection  
1717 established using an identity certificate shall be granted, regardless of the configuration of the  
1718 ACEs in the "/oic/sec/acl2" Resource, subject to the conditions in clause 10.4.2.

1719

## 1720 **9 Security Credential Management**

### 1721 **9.1 Preamble**

1722 This clause provides an overview of the credential types in OCF, along with details of credential  
1723 use, provisioning and ongoing management.

### 1724 **9.2 Credential Lifecycle**

#### 1725 **9.2.1 Credential Lifecycle General**

1726 OCF credential lifecycle has the following phases: (1) creation, (2) deletion, (3) refresh and (4)  
1727 revocation.

#### 1728 **9.2.2 Creation**

1729 The CMS can provision credentials to the credential Resource onto the Device. The Device shall  
1730 verify the CMS is authorized by matching the rowneruuid Property of the "/oic/sec/cred" Resource  
1731 to the DeviceID of the credential the CMS used to establish the secure connection.

1732 Credential Resources created using a CMS may involve specialized credential issuance protocols  
1733 and messages. These may involve the use of public key infrastructure (PKI) such as a certificate  
1734 authority (CA), symmetric key management such as a key distribution centre (KDC) or as part of a  
1735 provisioning action by a DOTS, CMS or AMS.

#### 1736 **9.2.3 Deletion**

1737 The CMS can delete credentials from the credential Resource. The Device (e.g. the Device where  
1738 the credential Resource is hosted) should delete credential Resources that have expired.

1739 An expired credential Resource may be deleted to manage memory and storage space.

1740 Deletion in OCF key management is equivalent to credential suspension.

#### 1741 **9.2.4 Refresh**

1742 Credential refresh may be performed before it expires. The CMS performs credential refresh.

1743 The "/oic/sec/cred" Resource supports expiry using the Period Property. Credential refresh may be  
1744 applied when a credential is about to expire or is about to exceed a maximum threshold for bytes  
1745 encrypted.

1746 A credential refresh method specifies the options available when performing key refresh. The  
1747 Period Property informs when the credential should expire. The Device may proactively obtain a  
1748 new credential using a credential refresh method using current unexpired credentials to refresh the  
1749 existing credential. If the Device does not have an internal time source, the current time should be  
1750 obtained from a CMS at regular intervals.

1751 If the onboarding established credentials are allowed to expire the DOTS shall re-onboard the  
1752 Device to re-apply device owner transfer steps.

1753 All Devices shall support at least one credential refresh method.

#### 1754 **9.2.5 Revocation**

1755 Credentials issued by a CMS may be equipped with revocation capabilities. In situations where the  
1756 revocation method involves provisioning of a revocation object that identifies a credential that is to  
1757 be revoked prior to its normal expiration period, a credential Resource is created containing the  
1758 revocation information that supersedes the originally issued credential. The revocation object  
1759 expiration should match that of the revoked credential so that the revocation object is cleaned up  
1760 upon expiry.

1761 It is conceptually reasonable to consider revocation applying to a credential or to a Device. Device  
1762 revocation asserts all credentials associated with the revoked Device should be considered for  
1763 revocation. Device revocation is necessary when a Device is lost, stolen or compromised. Deletion  
1764 of credentials on a revoked Device might not be possible or reliable.

### 1765 **9.3 Credential Types**

#### 1766 **9.3.1 Preamble**

1767 The "/oic/sec/cred" Resource maintains a credential type Property that supports several  
1768 cryptographic keys and other information used for authentication and data protection. The  
1769 credential types supported include symmetric pair-wise key, group symmetric group key,  
1770 asymmetric signing key, asymmetric signing key with certificate and shared-secret (i.e. PIN or  
1771 password). The Device shall always support symmetric pair-wise key and asymmetric signing key  
1772 with certificate credential types. Other credential types are optional.

#### 1773 **9.3.2 Pair-wise Symmetric Key Credentials**

1774 The CMS shall provision exactly one other pair-wise symmetric credential to a peer Device. The  
1775 CMS should not store pair-wise symmetric keys it provisions to managed Devices.

1776 Pair-wise keys could be established through ad-hoc key agreement protocols.

1777 The "PrivateData" Property in the "/oic/sec/cred" Resource contains the symmetric key.

1778 The "PublicData" Property may contain a token encrypted to the peer Device containing the pair-  
1779 wise key.

1780 The "OptionalData" Property may contain revocation status.

1781 The Device implementer should apply hardened key storage techniques that ensure the  
1782 "PrivateData" remains private.

1783 The Device implementer should apply appropriate integrity, confidentiality and access protection  
1784 of the "/oic/sec/cred", "/oic/sec/roles", "/oic/sec/csr" Resources to prevent unauthorized  
1785 modifications.

#### 1786 **9.3.3 Group Symmetric Key Credentials**

1787 Group keys are symmetric keys shared among a group of Devices (3 or more). Group keys are  
1788 used for efficient sharing of data among group participants.

1789 Group keys do not provide authentication of Devices but only establish membership in a group.

1790 The CMS shall provision group symmetric key credentials to the group members. The CMS  
1791 maintains the group memberships.

1792 The "PrivateData" Property in the "/oic/sec/cred" Resource contains the symmetric key.

1793 The "PublicData" Property may contain the group name.

1794 The "OptionalData" Property may contain revocation status.

1795 The Device implementer should apply hardened key storage techniques that ensure the  
1796 "PrivateData" remains private.

1797 The Device implementer should apply appropriate integrity, confidentiality and access protection  
1798 of the "/oic/sec/cred", "/oic/sec/roles", "/oic/sec/csr" Resources to prevent unauthorized  
1799 modifications.

1800 **9.3.4 Asymmetric Authentication Key Credentials**

1801 **9.3.4.1 Asymmetric Authentication Key Credentials General**

1802 Asymmetric authentication key credentials contain either a public and private key pair or only a  
1803 public key. The private key is used to sign Device authentication challenges. The public key is used  
1804 to verify a device authentication challenge-response.

1805 The "PrivateData" Property in the "/oic/sec/cred" Resource contains the private key.

1806 The "PublicData" Property contains the public key.

1807 The "OptionalData" Property may contain revocation status.

1808 The Device implementer should apply hardened key storage techniques that ensure the  
1809 "PrivateData" remains private.

1810 Devices should generate asymmetric authentication key pairs internally to ensure the private key  
1811 is only known by the Device. See 9.3.4.2 for when it is necessary to transport private key material  
1812 between Devices.

1813 The Device implementer should apply appropriate integrity, confidentiality and access protection  
1814 of the "/oic/sec/cred", "/oic/sec/roles", "/oic/sec/csr" Resources to prevent unauthorized  
1815 modifications.

1816 **9.3.4.2 External Creation of Asymmetric Authentication Key Credentials**

1817 Devices should employ industry-standard high-assurance techniques when allowing off-device key  
1818 pair creation and provisioning. Use of such key pairs should be minimized, particularly if the key  
1819 pair is immutable and cannot be changed or replaced after provisioning.

1820 When used as part of onboarding, these key pairs can be used to prove the Device possesses the  
1821 manufacturer-asserted properties in a certificate to convince a DOTS or a user to accept  
1822 onboarding the Device. See 7.3.3 for the OTM that uses such a certificate to authenticate the  
1823 Device, and then provisions new OCF Security Domain credentials for use.

1824 **9.3.5 Asymmetric Key Encryption Key Credentials**

1825 The asymmetric key-encryption-key (KEK) credentials are used to wrap symmetric keys when  
1826 distributing or storing the key.

1827 The "PrivateData" Property in the "/oic/sec/cred" Resource contains the private key.

1828 The "PublicData" Property contains the public key.

1829 The "OptionalData" Property may contain revocation status.

1830 The Device implementer should apply hardened key storage techniques that ensure the  
1831 "PrivateData" remains private.

1832 The Device implementer should apply appropriate integrity, confidentiality and access protection  
1833 of the "/oic/sec/cred", "/oic/sec/roles", "/oic/sec/csr" Resources to prevent unauthorized  
1834 modifications.

1835 **9.3.6 Certificate Credentials**

1836 Certificate credentials are asymmetric keys that are accompanied by a certificate issued by a CMS  
1837 or an external certificate authority (CA).

1838 A certificate enrolment protocol is used to obtain a certificate and establish proof-of-possession.

1839 The issued certificate is stored with the asymmetric key credential Resource.

1840 Other objects useful in managing certificate lifecycle such as certificate revocation status are  
1841 associated with the credential Resource.

1842 Either an asymmetric key credential Resource or a self-signed certificate credential is used to  
1843 terminate a path validation.

1844 The "PrivateData" Property in the "/oic/sec/cred" Resource contains the private key.

1845 The "PublicData" Property contains the issued certificate.

1846 The "OptionalData" Property may contain revocation status.

1847 The Device implementer should apply hardened key storage techniques that ensure the  
1848 PrivateData remains private.

1849 The Device implementer should apply appropriate integrity, confidentiality and access protection  
1850 of the "/oic/sec/cred", "/oic/sec/roles", "/oic/sec/csr" Resources to prevent unauthorized  
1851 modifications.

1852 **9.3.7 Password Credentials**

1853 The "PrivateData" Property in the "/oic/sec/cred" Resource contains the PIN, password and other  
1854 values useful for changing and verifying the password.

1855 The "PublicData" Property may contain the user or account name if applicable.

1856 The "OptionalData" Property may contain revocation status.

1857 The Device implementer should apply hardened key storage techniques that ensure the  
1858 "PrivateData" remains private.

1859 The Device implementer should apply appropriate integrity, confidentiality and access protection  
1860 of the "/oic/sec/cred", "/oic/sec/roles", "/oic/sec/csr" Resources to prevent unauthorized  
1861 modifications.

1862 **9.4 Certificate Based Key Management**

1863 **9.4.1 Overview**

1864 To achieve authentication and transport security during communications in OCF Security Domain,  
1865 certificates containing public keys of communicating parties and private keys can be used.

1866 The certificate and private key may be issued by a local or remote certificate authority (CA).

1867 The OCF certificate format is a subset of X.509 format, only elliptic curve algorithm and PEM  
1868 encoding format are allowed, most of optional fields in X.509 are not supported so that the format  
1869 intends to meet the constrained Device's requirement.

1870 The CMS manages the certificate lifecycle for certificates it issues. The DOTS assigns a CMS to a  
1871 Device when it is newly onboarded.

1872 **9.4.2 X.509 Digital Certificate Profiles**

1873 **9.4.2.1 Digital Certificate Profile General**

1874 An OCF certificate format is a subset of X.509 format (version 3 or above) as defined in  
1875 IETF RFC 5280.

1876 This clause develops a profile to facilitate the use of X.509 certificates within OCF applications for  
 1877 those communities wishing to make use of X.509 technology. The X.509 v3 certificate format is  
 1878 described in detail, with additional information regarding the format and semantics of OCF specific  
 1879 extension(s). The supported standard certificate extensions are also listed.

1880 Certificate Format: The OCF certificate profile is derived from IETF RFC 5280. However, this  
 1881 document does not support the "issuerUniqueID" and "subjectUniqueID" fields which are  
 1882 deprecated and shall not be used in the context of OCF. If these fields are present in a certificate,  
 1883 compliant entities shall ignore their contents.

1884 Certificate Encoding: Conforming entities shall use the Privacy-Enhanced Mail (PEM) to encode  
 1885 certificates.

1886 Certificates Hierarchy and Crypto Parameters. OCF supports a three-tier hierarchy for its Public  
 1887 Key Infrastructure (i.e., a Root CA, an Intermediate CA, and EE certificates). OCF accredited CAs  
 1888 SHALL use Elliptic Curve Cryptography (ECC) keys (secp256r1 – OID:1.2.840.10045.3.1.7) and  
 1889 use the ecdsaWithSHA256 (OID:1.2.840.10045.4.3.2) algorithm for certificate signatures. Elliptic  
 1890 Curve Cryptography public keys shall be encoded using uncompressed Elliptic Curve points.

1891 The following clauses specify the supported standard and custom extensions for the OCF  
 1892 certificates profile.

1893 **9.4.2.2 Certificate Profile and Fields**

1894 **9.4.2.2.1 Root CA Certificate Profile**

1895 Table 8 describes X.509 v1 fields required for Root CA Certificates.

1896 **Table 8 – X.509 v1 fields for Root CA Certificates**

V1 Field	Value / Remarks
signatureAlgorithm	ecdsa-with-SHA256 (OID: 1.2.840.10045.4.3.2)
Version	v3 (value is 2)
SerialNumber	SHALL be a positive integer, unique among all certificates issued by a given CA
Issuer	SHALL match the Subject field
Subject	SHALL match the Issuer field
notBefore	The time at which the Root CA Certificate was generated. See 10.4.5 for details around IETF RFC 5280-compliant validity field formatting.
notAfter	No stipulation for expiry date. See 10.4.5 for details around IETF RFC 5280-compliant validity field formatting.
Subject Public Key Info	id-ecPublicKey (OID: 1.2.840.10045.2.1) secp256r1 (OID:1.2.840.10045.3.1.7) Elliptic Curve Cryptography public keys shall be encoded using uncompressed Elliptic Curve points.

1897 Table 9 describes X.509 v3 extensions required for Root CA Certificates.

1898 **Table 9 - X.509 v3 extensions for Root CA Certificates**

Extension	Required/Optional	Criticality	Value / Remarks
authorityKeyIdentifier	OPTIONAL	Non-critical	N/A
subjectKeyIdentifier	OPTIONAL	Non-critical	N/A



keyUsage	REQUIRED	Critical	keyCertSign (5) & cRLSign (6) bits shall be enabled. digitalSignature(0) bit may be enabled. All other bits shall not be enabled.
basicConstraints	REQUIRED	Critical	cA = TRUE pathLenConstraint = not present (unlimited)

1899 **9.4.2.2.2 Intermediate CA Certificate Profile**

1900 Table 10 describes X.509 v1 fields required for Intermediate CA Certificates.

1901 **Table 10 - X.509 v1 fields for Intermediate CA Certificates**

V1 Field	Value / Remarks
signatureAlgorithm	ecdsa-with-SHA256 (OID: 1.2.840.10045.4.3.2)
Version	v3 (value is 2)
SerialNumber	SHALL be a positive integer, unique among all certificates issued by Root CA
Issuer	SHALL match the Subject field of the issuing Root CA
Subject	(no stipulation)
notBefore	The time at which the Intermediate CA Certificate was generated. See clause 10.4.5 for details around IETF RFC 5280-compliant validity field formatting.
notAfter	No stipulation for expiry date. See clause 10.4.5 for details around IETF RFC 5280-compliant validity field formatting.
Subject Public Key Info	id-ecPublicKey (OID: 1.2.840.10045.2.1) secp256r1 (OID: 1.2.840.10045.3.1.7) Elliptic Curve Cryptography public keys shall be encoded using uncompressed Elliptic Curve points.

1902 Table 11 describes X.509 v3 extensions required for Intermediate CA Certificates.

1903 **Table 11 – X.509 v3 extensions for Intermediate CA Certificates**

Extension	Required/Optional	Criticality	Value / Remarks
authorityKeyIdentifier	OPTIONAL	Non-critical	N/A
subjectKeyIdentifier	OPTIONAL	Non-critical	N/A
keyUsage	REQUIRED	Critical	keyCertSign (5) & cRLSign (6) bits shall be enabled. digitalSignature (0) bit may be enabled All other bits shall not be enabled.
basicConstraints	REQUIRED	Critical	cA = TRUE pathLenConstraint = 0 (can only sign End-Entity certs)
certificatePolicies	OPTIONAL	Non-critical	(no stipulation)
cRLDistributionPoints	OPTIONAL	Non-critical	1 or more URIs where the Certificate Revocation List

			(CRL) from the Root can be obtained.
authorityInformationAccess	OPTIONAL	Non-critical	OCSP URI – the URI of the Root CA's OCSP Responder

1904 **9.4.2.2.3 End-Entity Black Certificate Profile**

1905 Table 12 describes X.509 v1 fields required for End-Entity Certificates used for Black security  
 1906 profile.

1907 **Table 12 – X.509 v1 fields for End-Entity Certificates**

V1 Field	Value / Remarks
signatureAlgorithm	ecdsa-with-SHA256 (OID: 1.2.840.10045.4.3.2)
Version	v3 (value is 2)
SerialNumber	SHALL be a positive integer, unique among all certificates issued by the Intermediate CA
Issuer	SHALL match the Subject field of the issuing Intermediate CA
Subject	Subject DN shall include: o=OCF-verified device manufacturer organization name.  The Subject DN may include other attributes (e.g. cn, c, ou, etc.) with no stipulation by OCF.
notBefore	The time at which the End-Entity Certificate was generated. See clause 10.4.5 for details around IETF RFC 5280-compliant validity field formatting.
notAfter	No stipulation. See clause 10.4.5 for details around IETF RFC 5280-compliant validity field formatting.
Subject Public Key Info	id-ecPublicKey (OID: 1.2.840.10045.2.1) secp256r1 (OID:1.2.840.10045.3.1.7) Elliptic Curve Cryptography public keys shall be encoded using uncompressed Elliptic Curve points.

1908 Table 13 describes X.509 v3 extensions required for End-Entity Certificates.

1909 **Table 13 – X.509 v3 extensions for End-Entity Certificates**

Extension	Required/ Optional	Criticality	Value / Remarks
authorityKeyIdentifier	OPTIONAL	Non-critical	N/A
subjectKeyIdentifier	OPTIONAL	Non-critical	N/A
keyUsage	REQUIRED	Critical	digitalSignature (0) and keyAgreement(4) bits SHALL be the only bits enabled
basicConstraints	OPTIONAL	Non-Critical	cA = FALSE pathLenConstraint = not present
certificatePolicies	OPTIONAL	Non-critical	End-Entity certificates chaining to an OCF Root CA SHOULD contain at least one PolicyIdentifierId set to

			<p>the OCF Certificate Policy OID – (1.3.6.1.4.1.51414.0.1.2) corresponding to the version of the OCF Certificate Policy under which it was issued.</p> <p>Additional manufacturer-specific CP OIDs may also be populated.</p>
extendedKeyUsage	REQUIRED	Non-critical	<p>The following extendedKeyUsage (EKU) OIDs SHALL both be present:</p> <ul style="list-style-type: none"> <li>• serverAuthentication - 1.3.6.1.5.5.7.3.1</li> <li>• clientAuthentication - 1.3.6.1.5.5.7.3.2</li> </ul> <p>Exactly ONE of the following OIDs SHALL be present:</p> <ul style="list-style-type: none"> <li>• Identity certificate - 1.3.6.1.4.1.44924.1.6</li> <li>• Role certificate - 1.3.6.1.4.1.44924.1.7</li> </ul> <p>End-Entity certificates SHALL NOT contain the anyExtendedKeyUsage OID (2.5.29.37.0)</p>
subjectAlternativeName	REQUIRED UNDER CERTAIN CONDITIONS	Non-critical	<p>The subjectAltName extension is used to encode one or more Role ID values in role certificates, binding the roles to the subject public key.</p> <p>When the extendedKeyUsage (EKU) extension contains the Identity Certificate OID (1.3.6.1.4.1.44924.1.6), the subjectAltName extension SHOULD NOT be present.</p> <p>If the EKU extension contains the Role Certificate OID (1.3.6.1.4.1.44924.1.7), the subjectAltName extension SHALL be present and populated as follows: Each GeneralName in the GeneralNames SEQUENCE which encodes a role shall be a directoryName, which is of type Name. Name is an X.501 Distinguished Name. Each Name shall contain exactly one CN (Common Name) component, and zero or one OU (Organizational Unit) components. The OU component, if present, shall specify the authority that defined the semantics of the role. If the OU component is absent, the certificate issuer has defined the role. The CN</p>

			component shall encode the role ID. Other GeneralName types in the SEQUENCE may be present, but shall not be interpreted as roles. The role, and authority shall be encoded as ASN.1 PrintableString type, the restricted character set [0-9a-z-A-z '()+, -./:=?].
cRLDistributionPoints	OPTIONAL	Non-critical	1 or more URIs where the Certificate Revocation List (CRL) from the Intermediate CA can be obtained.
authorityInformationAccess	OPTIONAL	Non-critical	OCSP URI – the URI of the Intermediate CA's OCSP Responder
OCF Compliance	OPTIONAL	Non-critical	See 9.4.2.2.4
Manufacturer Usage Description (MUD)	OPTIONAL	Non-critical	Contains a single Uniform Resource Locator (URL) that points to an on-line Manufacturer Usage Description concerning the certificate subject. See 9.4.2.2.5
OCF Security Claims	OPTIONAL	Non-critical	Contains a list of security claims above those required by this OCF Compliance version or Security Profile. See 9.4.2.2.6
OCF CPL Attributes	OPTIONAL	Non-critical	Contains the list of OCF Attributes used to perform OCF Certified Product List lookups

1910 **9.4.2.2.4 OCF Compliance X.509v3 Extension**

1911 The OCF Compliance Extension defines required parameters to correctly identify the type of Device,  
1912 its manufacturer, its OCF Version, and the Security Profile compliance of the device.

1913 The extension carries an "ocfVersion" field which provides the specific base version of the OCF  
1914 documents the device implements. The "ocfVersion" field shall contain a sequence of three integers  
1915 ("major", "minor", and "build"). For example, if an entity is certified to be compliant with OCF  
1916 specifications 1.3.2, then the "major", "minor", and "build" fields of the "ocfVersion" will be set to  
1917 "1", "3", and "2" respectively. The "ocfVersion" may be used by Security Profiles to denote  
1918 compliance to a specified base version of the OCF documents.

1919 The "securityProfile" field shall carry the ocfSecurityProfile OID(s) (clause 14.8.3) of one or more  
1920 supported Security Profiles associated with the certificate in string form (UTF-8). All Security  
1921 Profiles associated with the certificate should be identified by this field.

1922 The extension shall also carry two string fields (UTF-8): "DeviceName" and "deviceManufacturer".  
1923 The fields carry human-readable descriptions of the Device's name and manufacturer, respectively.

1924 The ASN.1 definition of the OCFCompliance extension (OID – 1.3.6.1.4.1.51414.1.0) is defined as  
1925 follows:

```
1926 id-OCF OBJECT IDENTIFIER ::= { iso(1) identified-organization(3) dod(6) internet(1)
1927     private(4) enterprise(1) OCF(51414) }
1928
1929     id-ocfX509Extensions OBJECT IDENTIFIER ::= { id-OCF 1 }
```

```

1930
1931     id-ocfCompliance OBJECT IDENTIFIER ::= { id-ocfX509Extensions 0 }
1932
1933 ocfVersion ::= SEQUENCE {
1934     major    INTEGER,
1935             --Major version number
1936     minor    INTEGER,
1937             --Minor version number
1938     build    INTEGER,
1939             --Build/Micro version number
1940 }
1941
1942 ocfCompliance ::= SEQUENCE {
1943     version          ocfVersion,
1944                     --Device/OCF version
1945     securityProfile  SEQUENCE SIZE (1..MAX) OF ocfSecurityProfileOID,
1946                     --Sequence of OCF Security Profile OID strings
1947                     --Clause 14.8.2 defines valid ocfSecurityProfileOIDs
1948     deviceName       UTF8String,
1949                     --Name of the device
1950     deviceManufacturer UTF8String,
1951                     --Human-Readable Manufacturer
1952                     --of the device
1953 }

```

#### 1954 **9.4.2.2.5 Manufacturer Usage Description (MUD) X.509v3 Extension**

1955 The goal of the Manufacturer Usage Description (MUD) extension is to provide a means for devices  
1956 to signal to the network the access and network functionality they require to properly function.  
1957 Access controls can be more easily achieved and deployed at scale when the MUD extension is  
1958 used.

1959 The MUD X.509 v3 extension is specified in IETF RFC 8520 with the full ASN.1 definition in section  
1960 11.

#### 1961 **9.4.2.2.6 OCF Security Claims X.509v3 Extension**

1962 The OCF Security Claims Extension defines a list of OIDs representing security claims that the  
1963 manufacturer/integrator is making as to the security posture of the device above those required by  
1964 the OCF Compliance version or that of the OCF Security Profile being indicated by the device.

1965 The purpose of this extension is to allow for programmatic evaluation of assertions made about  
1966 security to enable some platforms/policies/administrators to better understand what is being  
1967 onboarded or challenged.

1968 The ASN.1 definition of the OCF Security Claims extension (OID – 1.3.6.1.4.1.51414.1.1) is defined  
1969 as follows:

```

1970 id-OCF OBJECT IDENTIFIER ::= { iso(1) identified-organization(3) dod(6) internet(1)
1971                               private(4) enterprise(1) OCF(51414) }
1972
1973     id-ocfX509Extensions OBJECT IDENTIFIER ::= { id-OCF 1 }
1974
1975     id-ocfSecurityClaims OBJECT IDENTIFIER ::= { id-ocfX509Extensions 1 }
1976
1977         claim-secure-boot ::= ocfSecurityClaimsOID { id-ocfSecurityClaims 0 }
1978         --Device claims that the boot process follows a procedure trusted
1979         --by the firmware and the BIOS
1980
1981         claim-hw-backed-cred-storage ::= ocfSecurityClaimsOID { id-ocfSecurityClaims 1 }
1982         --Device claims that credentials are stored in a specialized hardware
1983         --protection environment such as a Trusted Platform Module (TPM) or

```

1984 --similar mechanism.

1985

1986 ocfSecurityClaimsOID ::= OBJECT IDENTIFIER

1987

1988 ocfSecurityClaims ::= SEQUENCE SIZE (1..MAX) of ocfSecurityClaimsOID

#### 1989 **9.4.2.2.7 OCF Certified Product List Attributes X.509v3 Extension**

1990 The OCF Certified Product List Extension defines required parameters to utilize the OCF  
1991 Compliance Management System Certified Product List (OCMS-CPL). This clause is only  
1992 applicable if you plan to utilize the OCMS-CPL. The OBT may make use of these attributes to verify  
1993 the compliance level of a device.

1994 The extension carries the OCF CPL Attributes: IANA Private Enterprise Number (PEN), Model and  
1995 Version.

1996 The 'cpl-at-IANAPen' IANA Private Enterprise Number (PEN) provides the manufacturer's unique  
1997 PEN established in the IANA PEN list located at: [https://www.iana.org/assignments/enterprise-](https://www.iana.org/assignments/enterprise-numbers)  
1998 [numbers](https://www.iana.org/assignments/enterprise-numbers). The 'cpl-at-IANAPen' field found in end-products shall be the same information as  
1999 reported during OCF Certification.

2000 The 'cpl-at-model' represents an OCF-Certified product's model name. The 'cpl-at-model' field  
2001 found in end-products shall be the same information as reported during OCF Certification.

2002 The 'cpl-at-version' represents an OCF-Certified product's version. The 'cpl-at-version' field found  
2003 in end-products shall be the same information as reported during OCF Certification.

2004 The ASN.1 definition of the OCF CPL Attributes extension (OID – 1.3.6.1.4.1.51414.1.2) is defined  
2005 as follows:

2006 id-OCF OBJECT IDENTIFIER ::= { iso(1) identified-organization(3) dod(6) internet(1)  
2007 private(4) enterprise(1) OCF(51414) }

2008

2009 id-ocfX509Extensions OBJECT IDENTIFIER ::= { id-OCF 1 }

2010

2011 id-ocfCPLAttributes OBJECT IDENTIFIER ::= { id-ocfX509Extensions 2 }

2012

2013 cpl-at-IANAPen ::= OBJECT IDENTIFIER { id-ocfCPLAttributes 0 }

2014 cpl-at-model ::= OBJECT IDENTIFIER { id-ocfCPLAttributes 1 }

2015 cpl-at-version ::= OBJECT IDENTIFIER { id-ocfCPLAttributes 2 }

2016

2017

2018 ocfCPLAttributes ::= SEQUENCE {  
2019 cpl-at-IANAPen UTF8String,  
2020 --Manufacturer's registered IANA Private Enterprise Number  
2021 cpl-at-model UTF8String,  
2022 --Device OCF Security Profile  
2023 cpl-at-version UTF8String  
2024 --Name of the device  
2025 }

#### 2026 **9.4.2.3 Supported Certificate Extensions**

2027 As these certificate extensions are a standard part of IETF RFC 5280, this document includes the  
2028 clause number from that RFC to include it by reference. Each extension is summarized here, and  
2029 any modifications to the RFC definition are listed. Devices MUST implement and understand the  
2030 extensions listed here; other extensions from the RFC are not included in this document and  
2031 therefore are not required. 10.4 describes what Devices must implement when validating certificate  
2032 chains, including processing of extensions, and actions to take when certain extensions are absent.

2033 – Authority Key Identifier (4.2.1.1)

2034 The Authority Key Identifier (AKI) extension provides a means of identifying the public key  
2035 corresponding to the private key used to sign a certificate. This document makes the following  
2036 modifications to the referenced definition of this extension:

2037 The "authorityCertIssuer" or "authorityCertSerialNumber" fields of the "AuthorityKeyIdentifier"  
2038 sequence are not permitted; only "keyIdentifier" is allowed. This results in the following  
2039 grammar definition:

```
2040 id-ce-authorityKeyIdentifier OBJECT IDENTIFIER ::= { id-ce 35 }
```

```
2041 AuthorityKeyIdentifier ::= SEQUENCE {  
2042     keyIdentifier [0] KeyIdentifier }
```

```
2043  
2044  
2045 KeyIdentifier ::= OCTET STRING
```

#### 2046 – Subject Key Identifier (4.2.1.2)

2047 The Subject Key Identifier (SKI) extension provides a means of identifying certificates that  
2048 contain a particular public key.

2049 This document makes the following modification to the referenced definition of this extension:

2050 Subject Key Identifiers SHOULD be derived from the public key contained in the certificate's  
2051 "SubjectPublicKeyInfo" field or a method that generates unique values. This document  
2052 RECOMMENDS the 256-bit SHA-2 hash of the value of the BIT STRING "subjectPublicKey"  
2053 (excluding the tag, length, and number of unused bits). Devices verifying certificate chains must  
2054 not assume any particular method of computing key identifiers, however, and must only base  
2055 matching AKI's and SKI's in certification path constructions on key identifiers seen in certificates.

#### 2056 – Subject Alternative Name

2057 If the EKU extension is present, and has the value XXXXXX, indicating that this is a role  
2058 certificate, the Subject Alternative Name (subjectAltName) extension shall be present and  
2059 interpreted as described below. When no EKU is present, or has another value, the  
2060 "subjectAltName" extension SHOULD be absent. The "subjectAltName" extension is used to  
2061 encode one or more Role ID values in role certificates, binding the roles to the subject public  
2062 key. The "subjectAltName" extension is defined in IETF RFC 5280 (See 4.2.1.6):

```
2063 id-ce-subjectAltName OBJECT IDENTIFIER ::= { id-ce 17 }
```

```
2064  
2065 SubjectAltName ::= GeneralNames
```

```
2066  
2067 GeneralNames ::= SEQUENCE SIZE (1..MAX) OF GeneralName
```

```
2068  
2069 GeneralName ::= CHOICE {  
2070     otherName [0] OtherName,  
2071     rfc5322Name [1] IA5String,  
2072     dNSName [2] IA5String,  
2073     x400Address [3] ORAddress,  
2074     directoryName [4] Name,  
2075     ediPartyName [5] EDIPartyName,  
2076     uniformResourceIdentifier [6] IA5String,  
2077     iPAddress [7] OCTET STRING,  
2078     registeredID [8] OBJECT IDENTIFIER }
```

```
2079  
2080 EDIPartyName ::= SEQUENCE {  
2081     nameAssigner [0] DirectoryString OPTIONAL,  
2082     partyName [1] DirectoryString }
```

2084 Each "GeneralName" in the "GeneralNames" SEQUENCE which encodes a role shall be a  
2085 "directoryName", which is of type Name. Name is an X.501 Distinguished Name. Each Name  
2086 shall contain exactly one CN (Common Name) component, and zero or one OU (Organizational  
2087 Unit) components. The OU component, if present, shall specify the authority that defined the

2088 semantics of the role. If the OU component is absent, the certificate issuer has defined the role.  
2089 The CN component shall encode the role ID. Other "GeneralName" types in the SEQUENCE  
2090 may be present, but shall not be interpreted as roles. Therefore, if the certificate issuer includes  
2091 non-role names in the "subjectAltName" extension, the extension should not be marked critical.

2092 The role, and authority need to be encoded as ASN.1 "PrintableString" type, the restricted  
2093 character set [0-9a-z-A-z '()+, -./:=?].

2094 – Key Usage (4.2.1.3)

2095 The key usage extension defines the purpose (e.g., encipherment, signature, certificate signing)  
2096 of the key contained in the certificate. The usage restriction might be employed when a key that  
2097 could be used for more than one operation is to be restricted.

2098 This document does not modify the referenced definition of this extension.

2099 – Basic Constraints (4.2.1.9)

2100 The basic constraints extension identifies whether the subject of the certificate is a CA and the  
2101 maximum depth of valid certification paths that include this certificate. Without this extension,  
2102 a certificate cannot be an issuer of other certificates.

2103 This document does not modify the referenced definition of this extension.

2104 – Extended Key Usage (4.2.1.12)

2105  
2106 Extended Key Usage describes allowed purposes for which the certified public key may can be  
2107 used. When a Device receives a certificate, it determines the purpose based on the context of  
2108 the interaction in which the certificate is presented, and verifies the certificate can be used for  
2109 that purpose.

2110 This document makes the following modifications to the referenced definition of this extension:

2111 CAs SHOULD mark this extension as critical.

2112 CAs MUST NOT issue certificates with the anyExtendedKeyUsage OID (2.5.29.37.0).

2113

2114 The list of OCF-specific purposes and the assigned OIDs to represent them are:

2115 – Identity certificate 1.3.6.1.4.1.44924.1.6

2116 – Role certificate 1.3.6.1.4.1.44924.1.7

2117 **9.4.2.4 Cipher Suite for Authentication, Confidentiality and Integrity**

2118 OCF compliant entities shall support TLS version 1.2. Compliant entities shall support  
2119 TLS\_ECDHE\_ECDSA\_WITH\_AES\_128\_CCM\_8 cipher suite as defined in IETF RFC 7251 and may  
2120 support additional ciphers as defined in the TLS v1.2 specifications.

2121 **9.4.2.5 Encoding of Certificate**

2122 See 9.4.2 for details.

2123 **9.4.3 Certificate Revocation List (CRL) Profile [Deprecated]**

2124 This clause is intentionally left blank.

2125 **9.4.4 Resource Model**

2126 Device certificates and private keys are kept in "cred" Resource.

2127 The "cred" Resource contains the certificate information pertaining to the Device. The "PublicData"  
2128 Property holds the device certificate and CA certificate chain. "PrivateData" Property holds the  
2129 Device private key paired to the certificate. (See 13.3 for additional detail regarding the  
2130 "/oic/sec/cred" Resource).



2131 **9.4.5 Certificate Provisioning**

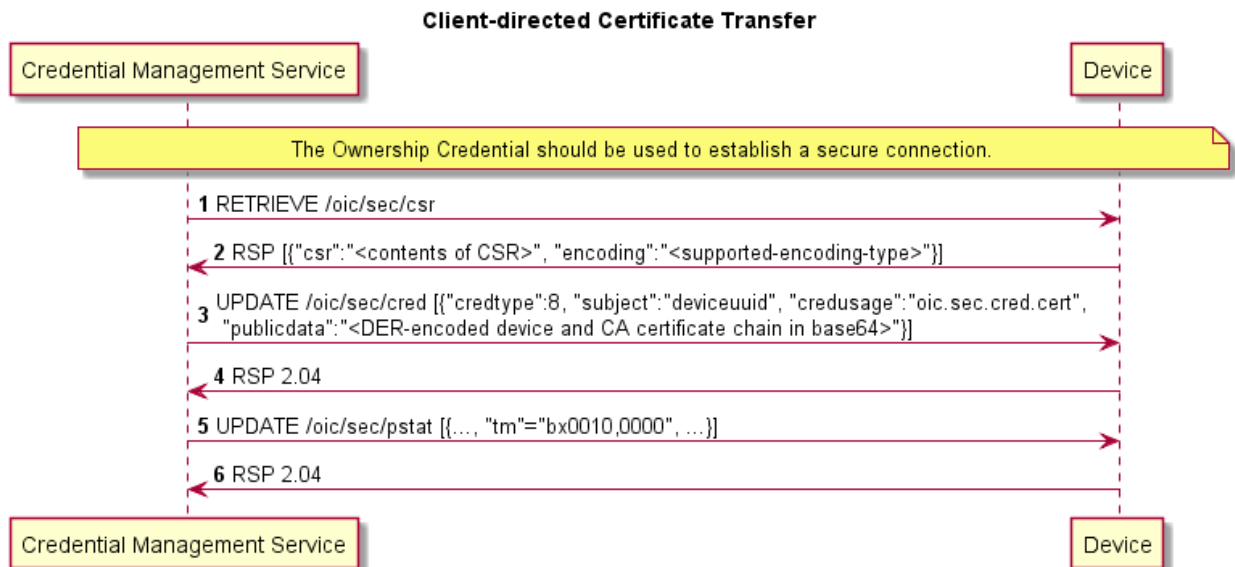
2132 The CMS (e.g. a hub or a smart phone) issues certificates for new Devices.

2133 The CA in the CMS retrieves a Device's public key and proof of possession of the private key,  
2134 generates a Device's certificate signed by this CA certificate, and then the CMS transfers them to  
2135 the Device including its CA certificate chain. Optionally, the CMS can also transfer one or more  
2136 role certificates, which shall have the format described in clause 9.4.2. The "subjectPublicKey" of  
2137 each role certificate shall match the "subjectPublicKey" in the Device certificate.

2138 In the sequence in Figure 19, the Certificate Signing Request (CSR) is defined by PKCS#10 in  
2139 IETF RFC 2986, and is included here by reference.

2140 The sequence flow of a certificate transfer for a Client-directed model is described in Figure 19.

- 2141 1) The CMS retrieves a CSR from the Device that requests a certificate. In this CSR, the Device  
2142 shall place its requested UUID into the subject and its public key in the "SubjectPublicKeyInfo".  
2143 The Device determines the public key to present; this may be an already-provisioned key it has  
2144 selected for use with authentication, or if none is present, it may generate a new key pair  
2145 internally and provide the public part. The key pair shall be compatible with the allowed  
2146 ciphersuites listed in 9.4.2.4 and 11.3.4, since the certificate will be restricted for use in OCF  
2147 authentication.
- 2148 2) 2) Alternatively, the CMS generates and provisions a private key and corresponding certificate  
2149 directly to the Device.
- 2150 3) The CMS transfers the issued certificate and CA chain to the designated Device using the same  
2151 credid, to maintain the association with the private key. The credential type ("oic.sec.cred")  
2152 used to transfer certificates in Figure 19 is also used to transfer role certificates, by including  
2153 multiple credentials in the POST from CMS to Device. Identity certificates shall be stored with  
2154 the credusage Property set to "oic.sec.cred.cert" and role certificates shall be stored with the  
2155 credusage Property set to "oic.sec.cred.rolecert".



2156

2157 **Figure 19 – Client-directed Certificate Transfer**

2158 **9.4.6 CRL Provisioning [Deprecated]**

2159 This clause is intentionally left blank.



## 2161 **10 Device Authentication**

### 2162 **10.1 Device Authentication General**

2163 When a Client is accessing a restricted Resource on a Server, the Server shall authenticate the  
2164 Client. Clients shall authenticate Servers while requesting access. Clients may also assert one or  
2165 more roles that the server can use in access control decisions. Roles may be asserted when the  
2166 Device authentication is done with certificates.

### 2167 **10.2 Device Authentication with Symmetric Key Credentials**

2168 When using symmetric keys to authenticate, the Server Device shall include the  
2169 ServerKeyExchange message and set `psk_identity_hint` to the Server's Device ID. The Client shall  
2170 validate that it has a credential with the Subject UUID set to the Server's Device ID, and a credential  
2171 type of PSK. If it does not, the Client shall respond with an `unknown_psk_identity` error or other  
2172 suitable error.

2173 If the Client finds a suitable PSK credential, it shall reply with a ClientKeyExchange message that  
2174 includes a `psk_identity` set to the Client's Device ID. The Server shall verify that it has a credential  
2175 with the matching Subject UUID and type. If it does not, the Server shall respond with an  
2176 `unknown_psk_identity` or other suitable error code. If it does, then it shall continue with the DTLS  
2177 protocol, and both Client and Server shall compute the resulting premaster secret.

### 2178 **10.3 Device Authentication with Raw Asymmetric Key Credentials**

2179 When using raw asymmetric keys to authenticate, the Client and the Server shall include a suitable  
2180 public key from a credential that is bound to their Device. Each Device shall verify that the provided  
2181 public key matches the `PublicData` field of a credential they have, and use the corresponding  
2182 Subject UUID of the credential to identify the peer Device.

### 2183 **10.4 Device Authentication with Certificates**

#### 2184 **10.4.1 Device Authentication with Certificates General**

2185 When using certificates to authenticate, the Client and Server shall each include their certificate  
2186 chain, as stored in the appropriate credential, as part of the selected authentication cipher suite.  
2187 Each Device shall validate the certificate chain presented by the peer Device. Each certificate  
2188 signature shall be verified until a public key is found within the `"/oic/sec/cred"` Resource with the  
2189 `"oic.sec.cred.trustca"` credusage. Credential Resource found in `"/oic/sec/cred"` is used to terminate  
2190 certificate path validation. Also, the validity period and revocation status should be checked for all  
2191 above certificates.

2192 A Device retrieves the Subject UUID from the Common Name component of the Subject Name  
2193 property of the End-Entity certificate which has the following format: `"uuid: X",` where X is  
2194 provisioned by the CMS to match the `"deviceuuid"` Property of the `"/oic/sec/doxm"` Resource. The  
2195 Device treats all requests arriving over a connection authenticated by this End-Entity certificate as  
2196 having originated from the Device with this Subject UUID. The Device shall use this Subject UUID  
2197 to match against the `"subjectuuid"` Property of the provisioned ACL entries to perform access  
2198 control checks.

2199 Devices must follow the certificate path validation algorithm in clause 6 of IETF RFC 5280. In  
2200 particular:

- 2201 – For all non-End-Entity certificates, Devices shall verify that the basic constraints extension is  
2202 present, and that the `cA` boolean in the extension is `TRUE`. If either is false, the certificate chain  
2203 **MUST** be rejected. If the `pathLenConstraint` field is present, Devices will confirm the number of  
2204 certificates between this certificate and the End-Entity certificate is less than or equal to  
2205 `pathLenConstraint`. In particular, if `pathLenConstraint` is zero, only an End-Entity certificate can  
2206 be issued by this certificate. If the `pathLenConstraint` field is absent, there is no limit to the  
2207 chain length.

- 2208 – For all non-End-Entity certificates, Devices shall verify that the key usage extension is present,  
2209 and that the keyCertSign bit is asserted.
- 2210 – Devices may use the Authority Key Identifier extension to quickly locate the issuing certificate.  
2211 Devices MUST NOT reject a certificate for lacking this extension, and must instead attempt  
2212 validation with the public keys of possible issuer certificates whose subject name equals the  
2213 issuer name of this certificate.
- 2214 – The End-Entity certificate of the chain shall be verified to contain an Extended Key Usage (EKU)  
2215 suitable to the purpose for which it is being presented. An End-Entity certificate which contains  
2216 no EKU extension is not valid for any purpose and must be rejected. Any certificate which  
2217 contains the anyExtendedKeyUsage OID (2.5.29.37.0) must be rejected, even if other valid  
2218 EKUs are also present.
- 2219 – Devices MUST verify "transitive EKU" for certificate chains. Issuer certificates (any certificate  
2220 that is not an End-Entity) in the chain MUST all be valid for the purpose for which the certificate  
2221 chain is being presented. An issuer certificate is valid for a purpose if it contains an EKU  
2222 extension and the EKU OID for that purpose is listed in the extension, OR it does not have an  
2223 EKU extension. An issuer certificate SHOULD contain an EKU extension and a complete list of  
2224 EKUs for the purposes for which it is authorized to issue certificates. An issuer certificate  
2225 without an EKU extension is valid for all purposes; this differs from End-Entity certificates  
2226 without an EKU extension.
- 2227 The list of purposes and their associated OIDs are defined in 9.4.2.3.

2228 If the Device does not recognize an extension, it must examine the "critical" field. If the field is  
2229 TRUE, the Device MUST reject the certificate. If the field is FALSE, the Device MUST treat the  
2230 certificate as if the extension were absent and proceed accordingly. This applies to all certificates  
2231 in a chain.

2232 NOTE Certificate revocation mechanisms are currently out of scope of this version of the document.

#### 2233 **10.4.2 Role Assertion with Certificates**

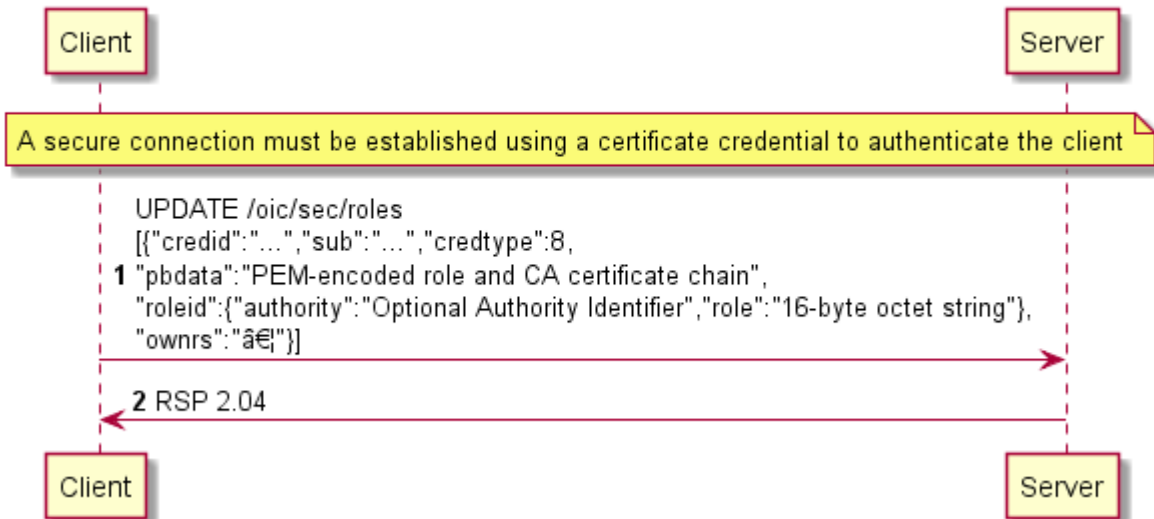
2234 This clause describes role assertion by a client to a server using a certificate role credential.

2235 Following authentication with a certificate, an OCF Client shall assert Roles by updating the  
2236 Server's "/oic/sec/roles" Resource with all the Role certificates it possesses, unless the device  
2237 manufacturer provides a vendor-specific mechanism for End User to select which roles to assert.  
2238 The Role credentials shall be certificate credentials and shall include a certificate chain. The Server  
2239 shall validate each certificate chain as specified in clause 10.3. Additionally, the public key in the  
2240 End-Entity certificate used for Device authentication shall be identical to the public key in all Role  
2241 (End-Entity) certificates. Also, the common name component of the subject name for both Role  
2242 certificates and identity certificates shall include a string of format "uuid:X" where X matches the  
2243 "deviceuuid" Property of the "oic.sec.doxm" Resource.

2244 Furthermore, a Client is prohibited from adding Role certificates for other Clients. The Server shall  
2245 reject Clients' request to add Role certificates if either (1) the request was received over an un-  
2246 secured connection or (2) the request was received over a secured connection but the public key  
2247 in the Role certificate does not match the public key in the identity certificate, which was used to  
2248 establish the secured connection.

2249 The Roles asserted are encoded in the subjectAltName extension in the certificate. The  
2250 "subjectAltName" field can have multiple values, allowing a single certificate to encode multiple  
2251 Roles that apply to the Client. The Server shall also check that the EKU extension of the Role  
2252 certificate(s) contains the value 1.3.6.1.4.1.44924.1.7 (see clause 9.4.2.2) indicating the certificate  
2253 may be used to assert Roles. Figure 20 describes how a Client Device asserts Roles to a Server.

## Asserting Certificate Role Credentials



2254

2255

**Figure 20 – Asserting a role with a certificate role credential.**

2256 Additional comments for Figure 20

- 2257 1) The response shall contain "204 No Content" to indicate success or 4xx to indicate an error. If  
 2258 the server does not support certificate credentials, it should return "501 Not Implemented"
- 2259 2) Roles asserted by the client may be kept for a duration chosen by the server. The duration shall  
 2260 not exceed the validity period of the role certificate.
- 2261 3) Servers should choose a nonzero duration to avoid the cost of frequent re-assertion of a role  
 2262 by a client. It is recommended that servers use the validity period of the certificate as a duration,  
 2263 effectively allowing the CMS to decide the duration.
- 2264 4) The format of the data sent in the create call shall be a list of credentials ("oic.sec.cred", see  
 2265 Table 19). They shall have "credtype" 8 (indicating certificates) and "PrivateData" field shall  
 2266 not be present. For fields that are duplicated in the "oic.sec.cred" object and the certificate, the  
 2267 value in the certificate shall be used for validation. For example, if the "Period" field is set in  
 2268 the credential, the server shall treat the validity period in the certificate as authoritative. Similar  
 2269 for the roleid data (authority, role).
- 2270 5) Certificates shall be encoded as in Figure 19 (PEM-encoded certificate chain).
- 2271 6) Clients may GET the "/oic/sec/roles" resource to determine the roles that have been previously  
 2272 asserted. An array of credential objects shall be returned. If there are no valid certificates  
 2273 corresponding to the currently connected and authenticated Client's identity, then an empty  
 2274 array (i.e. []) shall be returned.

### 2275 10.4.3 OCF PKI Roots

2276 This clause intentionally left empty.

### 2277 10.4.4 PKI Trust Store

2278 Each Device using a certificate chained to an OCF Root CA trust anchor SHALL securely store the  
 2279 OCF Root CA certificates in the "oic/sec/cred" resource and SHOULD physically store this resource  
 2280 in a hardened memory location where the certificates cannot be tampered with.

2281 **10.4.5 Path Validation and extension processing**

2282 Devices SHALL follow the certificate path validation algorithm in clause 6 of IETF RFC 5280. In  
2283 addition, the following are best practices and SHALL be adhered to by any OCF-compliant  
2284 application handling digital certificates

2285 – Validity Period checking

2286 OCF-compliant applications SHALL conform to IETF RFC 5280 clauses 4.1.2.5, 4.1.2.5.1, and  
2287 4.1.2.5.2 when processing the notBefore and notAfter fields in X.509 certificates. In addition,  
2288 for all certificates, the notAfter value SHALL NOT exceed the notAfter value of the issuing CA.

2289 – Revocation checking

2290 Relying applications SHOULD check the revocation status for all certificates.

2291 – basicConstraints

2292 For all Root and Intermediate Certificate Authority (CA) certificates, Devices SHALL verify that  
2293 the basicConstraints extension is present, flagged critical, and that the cA boolean value in the  
2294 extension is TRUE. If any of these are false, the certificate chain SHALL be rejected.

2295 If the pathLenConstraint field is present, Devices will confirm the number of certificates between  
2296 this certificate and the End-Entity certificate is less than or equal to pathLenConstraint. In  
2297 particular, if pathLenConstraint is zero, only an End-Entity certificate can be issued by this  
2298 certificate. If the pathLenConstraint field is absent, there is no limit to the chain length.

2299 For End-Entity certificates, if the basicConstraints extension is present, it SHALL be flagged  
2300 critical, SHALL have a cA boolean value of FALSE, and SHALL NOT contain a  
2301 pathLenConstraint ASN.1 sequence. An End-Entity certificate SHALL be rejected if a  
2302 pathLenConstraint ASN.1 sequence is either present with an Integer value, or present with a  
2303 null value.

2304 In order to facilitate future flexibility in OCF-compliant PKI implementations, all OCF-compliant  
2305 Root CA certificates SHALL NOT contain a pathLenConstraint. This allows additional tiers of  
2306 Intermediate CAs to be implemented in the future without changing the Root CA trust anchors,  
2307 should such a requirement emerge.

2308 – keyUsage

2309 For all certificates, Devices shall verify that the key usage extension is present and flagged  
2310 critical.

2311 For Root and Intermediate CA certificates, ONLY the keyCertSign(5) and crlSign(6) bits SHALL  
2312 be asserted.

2313 For End-Entity certificates, ONLY the digitalSignature(0) and keyAgreement(4) bits SHALL be  
2314 asserted.

2315 – extendedKeyUsage:

2316 Any End-Entity certificate containing the anyExtendedKeyUsage OID ("2.5.29.37.0") SHALL be  
2317 rejected.

2318 OIDs for serverAuthentication ("1.3.6.1.5.5.7.3.1") and clientAuthentication ("1.3.6.1.5.5.7.3.2")  
2319 are required for compatibility with various TLS implementations.

2320 At this time, an End-Entity certificate cannot be used for both Identity ("1.3.6.1.4.1.44924.1.6")  
2321 and Role ("1.3.6.1.4.1.44924.1.7") purposes. Therefore, exactly one of the two OIDs SHALL be  
2322 present and End-Entity certificates with ECU extensions containing both OIDs SHALL be  
2323 rejected.

2324 – certificatePolicies

2325 End-Entity certificates which chain to an OCF Root CA SHOULD contain at least one  
2326 PolicyIdentifierId set to the OCF Certificate Policy OID – ("1.3.6.1.4.1.51414.0.1.2")

2327 corresponding to the version of the OCF Certificate Policy under which it was issued. Additional  
2328 manufacturer-specific CP OIDs may also be populated.

2329 **10.5 Device Authentication with OCF Cloud – moved to OCF Cloud Security document**

2330 This clause is intentionally left blank.

2331

2332 **11 Message Integrity and Confidentiality**

2333 **11.1 Preamble**

2334 Secured communications between Clients and Servers are protected against eavesdropping,  
2335 tampering, or message replay, using security mechanisms that provide message confidentiality and  
2336 integrity.

2337 **11.2 Session Protection with DTLS**

2338 **11.2.1 DTLS Protection General**

2339 Devices shall support DTLS for secured communications as defined in IETF RFC 6347. Devices  
2340 using TCP shall support TLS v1.2 for secured communications as defined in IETF RFC 5246. See  
2341 11.3 for a list of required and optional cipher suites for message communication.

2342 OCF Devices MUST support (D)TLS version 1.2 or greater and MUST NOT support versions 1.1  
2343 or lower.

2344 Multicast session semantics are not yet defined in this version of the security document.

2345 **11.2.2 Unicast Session Semantics**

2346 For unicast messages between a Client and a Server, both Devices shall authenticate each other.  
2347 See clause 10 for details on Device Authentication.

2348 Secured unicast messages between a Client and a Server shall employ a cipher suite from 11.3.  
2349 The sending Device shall encrypt and authenticate messages as defined by the selected cipher  
2350 suite and the receiving Device shall verify and decrypt the messages before processing them.

2351 **11.2.3 Cloud Session Semantics – moved to OCF Cloud Security document**

2352 This clause is intentionally left blank.

2353 **11.3 Cipher Suites**

2354 **11.3.1 Cipher Suites General**

2355 The cipher suites allowed for use can vary depending on the context. This clause lists the cipher  
2356 suites allowed during ownership transfer and normal operation. The following RFCs provide  
2357 additional information about the cipher suites used in OCF.

2358 IETF RFC 4279: Specifies use of pre-shared keys (PSK) in (D)TLS

2359 IETF RFC 4492: Specifies use of elliptic curve cryptography in (D)TLS

2360 IETF RFC 5489: Specifies use of cipher suites that use elliptic curve Diffie-Hellman (ECDHE) and  
2361 PSKs

2362 IETF RFC 6655 and IETF RFC 7251: Specifies AES-CCM mode cipher suites, with ECDHE

2363 **11.3.2 Cipher Suites for Device Ownership Transfer**

2364 **11.3.2.1 Just Works Method Cipher Suites**

2365 The Just Works OTM may use the following (D)TLS cipher suites.

2366 TLS\_ECDH\_ANON\_WITH\_AES\_128\_CBC\_SHA256

2367 All Devices supporting Just Works OTM shall implement:

2368 TLS\_ECDH\_ANON\_WITH\_AES\_128\_CBC\_SHA256 (with the value 0xFF00)



2369 **11.3.2.2 Random PIN Method Cipher Suites**

2370 The Random PIN Based OTM may use the following (D)TLS cipher suites.

2371 TLS\_ECDHE\_PSK\_WITH\_AES\_128\_CBC\_SHA256

2372 All Devices supporting Random Pin Based OTM shall implement:

2373 TLS\_ECDHE\_PSK\_WITH\_AES\_128\_CBC\_SHA256

2374 **11.3.2.3 Certificate Method Cipher Suites**

2375 The Manufacturer Certificate Based OTM may use the following (D)TLS cipher suites.

2376 TLS\_ECDHE\_ECDSA\_WITH\_AES\_128\_CCM\_8,

2377 TLS\_ECDHE\_ECDSA\_WITH\_AES\_256\_CCM\_8,

2378 TLS\_ECDHE\_ECDSA\_WITH\_AES\_128\_CCM,

2379 TLS\_ECDHE\_ECDSA\_WITH\_AES\_256\_CCM

2380 Using the following curve:

2381 secp256r1 (See IETF RFC 4492)

2382 All Devices supporting Manufacturer Certificate Based OTM shall implement:

2383 TLS\_ECDHE\_ECDSA\_WITH\_AES\_128\_CCM\_8

2384 Devices supporting Manufacturer Certificate Based OTM should implement:

2385 TLS\_ECDHE\_ECDSA\_WITH\_AES\_256\_CCM\_8,

2386 TLS\_ECDHE\_ECDSA\_WITH\_AES\_128\_CCM,

2387 TLS\_ECDHE\_ECDSA\_WITH\_AES\_256\_CCM

2388 **11.3.3 Cipher Suites for Symmetric Keys**

2389 The following cipher suites are defined for (D)TLS communication using PSKs:

2390 TLS\_ECDHE\_PSK\_WITH\_AES\_128\_CBC\_SHA256,

2391 TLS\_PSK\_WITH\_AES\_128\_CCM\_8, (\* 8 OCTET Authentication tag \*)

2392 TLS\_PSK\_WITH\_AES\_256\_CCM\_8,

2393 TLS\_PSK\_WITH\_AES\_128\_CCM, (\* 16 OCTET Authentication tag \*)

2394 TLS\_PSK\_WITH\_AES\_256\_CCM,

2395 All CCM based cipher suites also use HMAC-SHA-256 for authentication.

2396 All Devices shall implement the following:

2397 TLS\_ECDHE\_PSK\_WITH\_AES\_128\_CBC\_SHA256,

2398

2399 Devices should implement the following:

2400 TLS\_ECDHE\_PSK\_WITH\_AES\_128\_CBC\_SHA256,

2401 TLS\_PSK\_WITH\_AES\_128\_CCM\_8,

2402 TLS\_PSK\_WITH\_AES\_256\_CCM\_8,

2403 TLS\_PSK\_WITH\_AES\_128\_CCM,

2404 TLS\_PSK\_WITH\_AES\_256\_CCM

2405 **11.3.4 Cipher Suites for Asymmetric Credentials**

2406 The following cipher suites are defined for (D)TLS communication with asymmetric keys or  
2407 certificates:

2408 TLS\_ECDHE\_ECDSA\_WITH\_AES\_128\_CCM\_8,

2409 TLS\_ECDHE\_ECDSA\_WITH\_AES\_256\_CCM\_8,

2410 TLS\_ECDHE\_ECDSA\_WITH\_AES\_128\_CCM,

2411 TLS\_ECDHE\_ECDSA\_WITH\_AES\_256\_CCM

2412 Using the following curve:

2413 secp256r1 (See IETF RFC 4492)

2414 All Devices supporting Asymmetric Credentials shall implement:

2415 TLS\_ECDHE\_ECDSA\_WITH\_AES\_128\_CCM\_8

2416 All Devices supporting Asymmetric Credentials should implement:

2417 TLS\_ECDHE\_ECDSA\_WITH\_AES\_256\_CCM\_8,

2418 TLS\_ECDHE\_ECDSA\_WITH\_AES\_128\_CCM,

2419 TLS\_ECDHE\_ECDSA\_WITH\_AES\_256\_CCM

2420 **11.3.5 Cipher suites for OCF Cloud Credentials – moved to OCF Cloud Security document**

2421 This clause is intentionally left blank.

2422

2423 **12 Access Control**

2424 **12.1 ACL Generation and Management**

2425 This clause intentionally left empty.

2426 **12.2 ACL Evaluation and Enforcement**

2427 **12.2.1 ACL Evaluation and Enforcement General**

2428 The Server enforces access control over application Resources before exposing them to the  
2429 requestor. The Security Layer in the Server authenticates the requestor when access is received  
2430 via the secure port. Authenticated requestors, known as the "subject" can be used to match ACL  
2431 entries that specify the requestor's identity, role or may match authenticated requestors using a  
2432 subject wildcard.

2433 If the request arrives over the unsecured port, the only ACL policies allowed are those that use a  
2434 subject wildcard match of anonymous requestors.

2435 Access is denied if a requested Resource is not matched by an ACL entry.

2436 NOTE There are documented exceptions pertaining to Device onboarding where access to Security Virtual Resources  
2437 may be granted prior to provisioning of ACL Resources.

2438 The second generation ACL (i.e. "/oic/sec/acl2") contains an array of Access Control Entries (ACE2)  
2439 that employ a Resource matching algorithm that uses an array of Resource references to match  
2440 Resources to which the ACE2 access policy applies. Matching consists of comparing the values of  
2441 the ACE2 "resources" Property (see clause 13) to the requested Resource. Resources are matched  
2442 in two ways:

- 2443 1) host reference ("href")
- 2444 2) resource wildcard ("wc").

2445 **12.2.2 Host Reference Matching**

2446 When present in an ACE2 matching element, the Host Reference (href) Property shall be used for  
2447 Resource matching.

2448 – The href Property shall be used to find an exact match of the Resource name if present.

2449 **12.2.3 Resource Wildcard Matching**

2450 When present, a wildcard ("wc") expression shall be used to match multiple Resources using a  
2451 wildcard Property contained in the "oic.sec.ace2.resource-ref" structure.

2452 A wildcard expression may be used to match multiple Resources using a wildcard Property  
2453 contained in the "oic.sec.ace2.resource-ref" structure. The wildcard matching strings are defined  
2454 in Table 14.

2455 **Table 14 – ACE2 Wildcard Matching Strings Description**

String	Description
"+"	Shall match all Discoverable Non-Configuration Resources which expose at least one Secure OCF Endpoint.
"_"	Shall match all Discoverable Non-Configuration Resources which expose at least one Unsecure OCF Endpoint.
"**"	Shall match all Non-Configuration Resources.

2456 NOTE Discoverable resources appear in the "/oic/res" Resource, while non-discoverable resources may appear in other  
2457 collection resources but do not appear in the /res collection.

2458 **12.2.4 Multiple Criteria Matching**

2459 If the ACE2 "resources" Property contains multiple entries, then a logical OR shall be applied for  
2460 each array element. For example, if a first array element of the "resources" Property contains  
2461 "href="/a/light" and the second array element of the "resources" Property contains "href="/a/led",  
2462 then Resources that match either of the two "href" criteria shall be included in the set of matched  
2463 Resources.

2464 Example 1 JSON for Resource matching

```
2465 {  
2466 //Matches Resources named "/x/door1" or "/x/door2"  
2467 "resources": [  
2468   {  
2469     "href": "/x/door1"  
2470   },  
2471   {  
2472     "href": "/x/door2"  
2473   },  
2474 ]  
2475 }
```

2476 Example 2 JSON for Resource matching

```
2477 {  
2478 // Matches all Resources  
2479 "resources": [  
2480   {  
2481     "wc": "*"   
2482   }  
2483 ]  
2484 }
```

2485 **12.2.5 Subject Matching using Wildcards**

2486 When the ACE subject is specified as the wildcard string "\*" any requestor is matched. The OCF  
2487 server may authenticate the OCF client, but is not required to.

2488 Examples: JSON for subject wildcard matching

```
2489 //matches all subjects that have authenticated and confidentiality protections in place.  
2490 "subject" : {  
2491   "conntype" : "auth-crypt"  
2492 }  
2493 //matches all subjects that have NOT authenticated and have NO confidentiality protections in place.  
2494 "subject" : {  
2495   "conntype" : "anon-clear"  
2496 }
```

2497 **12.2.6 Subject Matching using Roles**

2498 When the ACE subject is specified as a role, a requestor shall be matched if either:

- 2499 1) The requestor authenticated with a symmetric key credential, and the role is present in the  
2500 "roleid" Property of the credential's entry in the "credential" Resource, or

2501 2) The requestor authenticated with a certificate, and a valid role certificate is present in the roles  
2502 resource with the requestor's certificate's public key at the time of evaluation. Validating role  
2503 certificates is defined in 10.3.1.

## 2504 **12.2.7 ACL Evaluation**

### 2505 **12.2.7.1 ACE2 matching algorithm**

2506 The OCF Server shall apply an ACE2 matching algorithm that matches in the following sequence:

- 2507 1) The local "/oic/sec/acl2" Resource contributes its ACE2 entries for matching.
- 2508 2) Access shall be granted when all these criteria are met:
  - 2509 a) The requestor is matched by the ACE2 "subject" Property.
  - 2510 b) The requested Resource is matched by the ACE2 "resources" Property and the requested  
2511 Resource shall exist on the local Server.
  - 2512 c) The "period" Property constraint shall be satisfied.
  - 2513 d) The "permission" Property constraint shall be applied.

2514 If multiple ACE2 entries match the Resource request, the union of permissions, for all matching  
2515 ACEs, defines the effective permission granted. E.g. If Perm1=CR---; Perm2=--UDN; Then UNION  
2516 (Perm1, Perm2)=CRUDN.

2517 The Server shall enforce access based on the effective permissions granted.

2518 Batch requests to Resource containing Links require additional considerations when accessing the  
2519 linked Resources. ACL considerations for batch request to the Atomic Measurement Resource  
2520 Type are provided in clause 12.2.7.2. ACL considerations for batch request to the Collection  
2521 Resource Type are provided in clause 12.2.7.3.

2522 Clause 12.2.7.4 provides ACL considerations when a new Resource is created on a Server in  
2523 response to a CREATE request.

### 2524 **12.2.7.2 (Currently blank)**

2525 This clause intentionally left empty.

### 2526 **12.2.7.3 ACL considerations for a batch OCF Interface request to a Collection**

2527 This clause addresses the additional authorization processes which take place when a Server  
2528 receives a batch OCF Interface request from a Client to a Collection hosted on that Server,  
2529 assuming there is an ACE matching the Collection which permits the original Client request. For  
2530 the purposes of this clause, the Server hosting this Collection is called the "Collection host". The  
2531 additional authorization process is dependent on whether the linked Resource is hosted on the  
2532 Collection host or the linked Resource is hosted on another Server:

- 2533 – For each generated request to a linked Resource hosted on the Collection host, the Collection  
2534 host shall apply the ACE2 matching algorithm in clause 12.2.7.1 to determine whether the linked  
2535 Resource is permitted to process the generated request, with the following clarifications:
  - 2536 – The requestor in clause 12.2.7.1 shall be the Client which sent the original Client request.
  - 2537 – The requested Resource in clause 12.2.7.1 shall be the linked Resource, which shall be  
2538 matched using at least one of:
    - 2539 – a Resource Wildcard matching the linked Resource, or
    - 2540 – an exact match of the local path of the linked Resource with a "href" Property in the  
2541 "resources" array in the ACE2.
    - 2542 – an exact match of the full URI of the linked Resource with a "href" Property in the  
2543 "resources" array in the ACE2.

2544 NOTE The full URI of a linked Resource is obtained by concatenating the "anchor" Property of the Link, if present, and  
2545 the "href" Property of the Link. The local path can then be determined from the full URI.

2546 If the linked Resource is not permitted to process the generated request, then the Collection host  
2547 shall treat such cases as a linked Resource which cannot process the request when composing the  
2548 aggregated response to the original Client Request, as specified for the batch OCF Interface in the  
2549 ISO/IEC 30118-1:2018.

#### 2550 **12.2.7.4 ACL Considerations on creation of a new Resource**

2551 When a new Resource is created on a Server in response to a CREATE request, there might be  
2552 no ACEs permitting access to the newly created Resource. The present clause describes how the  
2553 Server autonomously modifies the "/oic/sec/acl2" Resource to provide some initial authorizations  
2554 for accessing the newly created Resource. The purpose of this autonomous modification is to avoid  
2555 relying on the AMS update the "/oic/sec/acl2" Resource after every new Resource is created.

2556 Subsequent to a Server creating a Collection inside another Collection in response to a CREATE  
2557 request from a Client, and prior to sending a response to the Client:

- 2558 – If there is an ACE with "subject" containing the UUID of the Client, and "permissions" exactly  
2559 matching the CREATE, RETRIEVE, UPDATE and DELETE operations, then the Server shall  
2560 autonomously add an "href" entry to "resources" with the URI of the newly created Collection.
- 2561 – Otherwise, the Server shall autonomously add an ACE with "subject" containing the UUID  
2562 of the Client, "resources" containing an "href" entry with the URI of the newly created  
2563 Collection, and "permissions" exactly matching the CREATE, RETRIEVE, UPDATE and  
2564 DELETE operations.

2565 Subsequent to a Server creating a non-Collection Resource inside another Collection in response  
2566 to a CREATE request from a Client, and prior to sending a response to the Client:

- 2567 – If there is an ACE with "subject" containing the UUID of the Client, and "permissions" exactly  
2568 matching the RETRIEVE, UPDATE and DELETE operations, then the Server shall  
2569 autonomously add an "href" entry to "resources" with the URI of the newly created Resource.
- 2570 – Otherwise, the Server shall autonomously add an ACE with "subject" containing the UUID  
2571 of the Client, "resources" containing an "href" entry with the URI of the newly created, and  
2572 "permissions" exactly matching the RETRIEVE, UPDATE and DELETE operations.

2573

2574 **13 Security Resources**

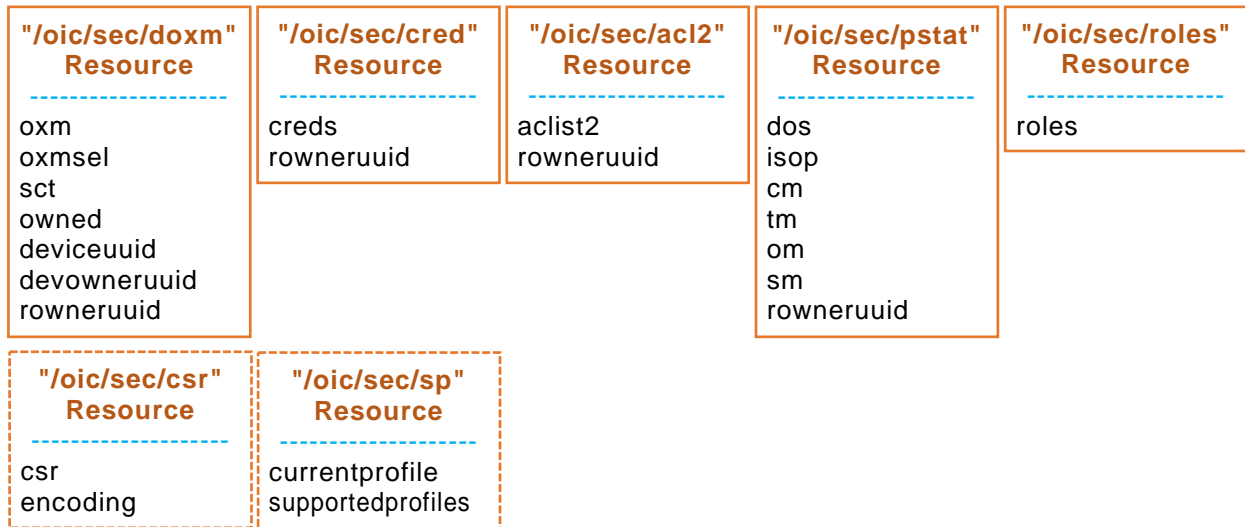
2575 **13.1 Security Resources General**

2576 OCF Security Resources are shown in Figure 21.

2577 "/oic/sec/cred" Resource and Properties are shown in Figure 22.

2578 "/oic/sec/acl2" Resource and Properties are shown in Figure 23.

2579



2580

**Figure 21 – OCF Security Resources**

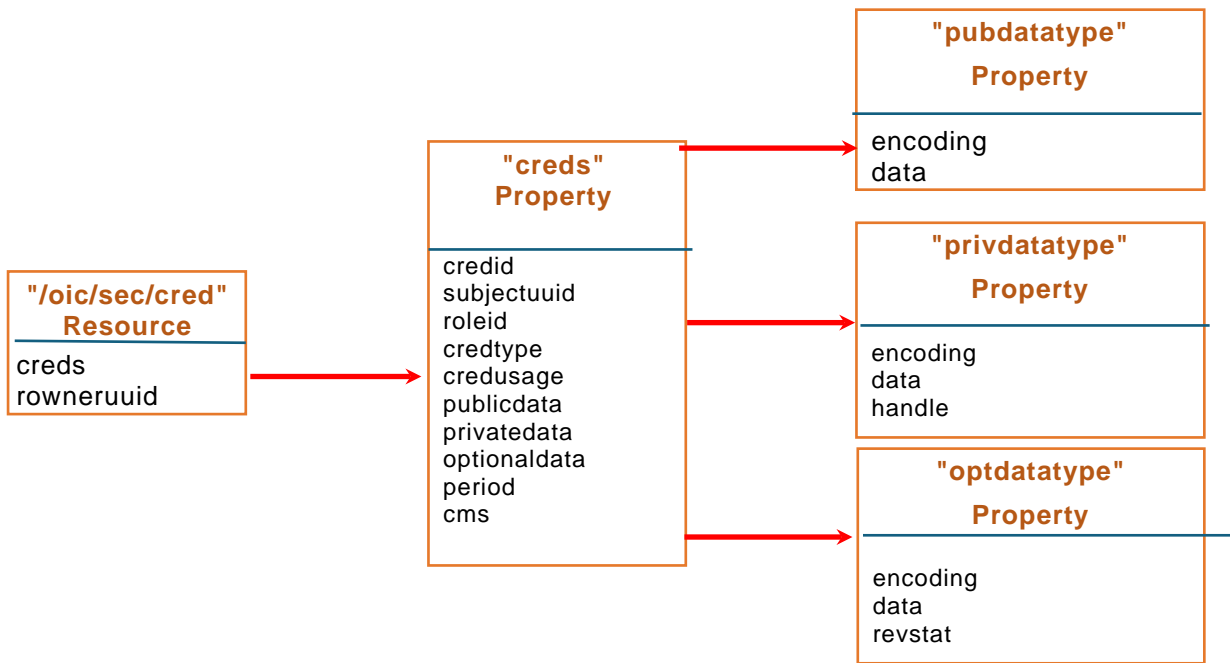


Figure 22 – "/oic/sec/cred" Resource and Properties

2581

2582

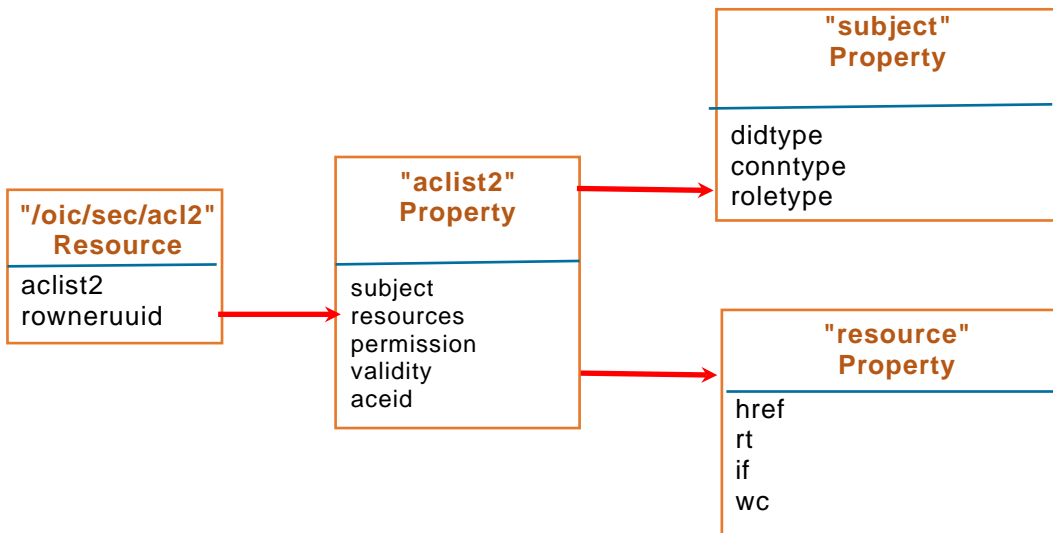


Figure 23 – "/oic/sec/acl2" Resource and Properties

2583

2584

## 2585 13.2 Device Owner Transfer Resource

### 2586 13.2.1 Device Owner Transfer Resource General

2587 The "/oic/sec/doxm" Resource contains the set of supported Device OTMs.



2588 Resource discovery processing respects the CRUDN constraints supplied as part of the security  
 2589 Resource definitions contained in this document.

2590 "/oic/sec/doxm" Resource is defined in Table 15.

2591 **Table 15 – Definition of the "/oic/sec/doxm" Resource**

Fixed URI	Resource Type Title	Resource Type ID ("rt" value)	OCF Interfaces	Description	Related Functional Interaction
/oic/sec/doxm	Device OTMs	oic.r.doxm	oic.if.baseline	Resource for supporting Device owner transfer	Configuration

2592 Table 16 defines the Properties of the "/oic/sec/doxm" Resource.

2593 **Table 16 – Properties of the "/oic/sec/doxm" Resource**

Property Title	Property Name	Value Type	Value Rule	Mandatory	Device State	Access Mode	Description
OTM	oxms	oic.sec.doxmtype	array	Yes		R	Value identifying the owner-transfer-method and the organization that defined the method.
OTM Selection	oxmsel	oic.sec.doxmtype	UINT16	Yes	RESET	R	Server shall set to (4) "oic.sec.oxm.self"
					RFOTM	RW	DOTS shall set to its selected DOTS and both parties execute the DOTS. After secure owner transfer session is established DOTS shall update the oxmsel again making it permanent. If the DOTS fails the Server shall transition device state to RESET.
					RFPRO	R	n/a
					RFNOP	R	n/a
					SRESET	R	n/a
Supported Credential Types	sct	oic.sec.credtype	bitmask	Yes		R	Identifies the types of credentials the Device supports. The Server sets this value at framework initialization after determining security capabilities.  The Device always supports symmetric pair-wise key and asymmetric signing key with certificate (bit positions 0x1 and 0x8 respectively). Other credential types are optional as per clause 9.3
Device Ownership Status	owned	Boolean	T F	Yes	RESET	R	Server shall set to FALSE.
					RFOTM	RW	DOTS shall set to TRUE after secure owner transfer session is established.
					RFPRO	R	n/a
					RFNOP	R	TRUE
					SRESET	R	TRUE
Device UUID	deviceuuid	String	oic.sec.didtype	Yes	RESET	R	No stipulation.
					RFOTM	RW	DOTS updates to a value it has selected after secure owner transfer session is established.
					RFPRO	R	n/a

					RFNOP	R	n/a
					SRESET	R	n/a
Device Owner Id	devowneruuid	String	uuid	Yes	RESET	R	Server shall set to the nil uuid value (e.g. "00000000-0000-0000-0000-000000000000" )
					RFOTM	RW	DOTS shall set value after secure owner transfer session is established.
					RFPRO	R	n/a
					RFNOP	R	n/a
					SRESET	R	n/a
Resource Owner Id	rowneruuid	String	uuid	Yes	RESET	R	Server shall set to the nil uuid value (e.g. "00000000-0000-0000-0000-000000000000" )
					RFOTM	RW	The DOTS shall configure the rowneruuid Property when a successful owner transfer session is established.
					RFPRO	R	n/a
					RFNOP	R	n/a
					SRESET	RW	The DOTS (referenced via devowneruuid Property) should verify and if needed, update the resource owner Property when a mutually authenticated secure session is established. If the rowneruuid does not refer to a valid DOTS device identifier the Server shall transition to RESET Device state.

2594 Table 17 defines the Properties of the "oic.sec.didtype".

2595 **Table 17 – Properties of the "oic.sec.didtype" type**

Property Title	Property Name	Value Type	Value Rule	Mandatory	Device State	Access Mode	Description
Device ID	uuid	String	uuid	Yes	RW	-	A uuid value

2596 The "oxms" Property contains a list of OTM where the entries appear in the order of preference.  
2597 This Property contains the higher priority methods appearing before the lower priority methods.  
2598 The DOTS queries this list at the time of onboarding and selects the most appropriate method.

2599 OTMs consist of two parts, a URI identifying the vendor or organization and the specific method.

```

2600 <DoxmType> ::= <NSS>
2601 <NSS> ::= <Identifier> | { {<NID>"."} <NameSpaceQualifier> "." } <Method>
2602 <NID> ::= <Vendor-or-Organization>
2603 <Identifier> ::= INTEGER
2604 <NameSpaceQualifier> ::= String
2605 <Method> ::= String
2606 <Vendor-Organization> ::= String

```

2607 When an OTM successfully completes, the "owned" Property is set to "1" (TRUE). Consequently,  
2608 subsequent attempts to take ownership of the Device will fail.

2609 There are four device identifiers:

- 2610 1) "deviceuuid" Property of "/oic/sec/doxm" Resource - random DOTS-provisioned value unique  
 2611 for a given security domain, used as a device identity for access control, mapped internally to  
 2612 a device-owned credential.
- 2613 2) "di" Property of "/oic/d" Resource - mirroring the value of "deviceuuid" Property of  
 2614 "/oic/sec/doxm" Resource.
- 2615 3) "piid" Property of "/oic/d" Resource - defined in ISO/IEC 30118-1:2018.
- 2616 4) "pi" Property of "/oic/p" Resource - defined in ISO/IEC 30118-1:2018.

2617 **13.2.2 OCF defined OTMs**

2618 Table 18 defines the Properties of the "oic.sec.doxmtype".

2619 **Table 18 – Properties of the "oic.sec.doxmtype" type**

Value Type Name	Value Type URN (optional)	Enumeration Value (mandatory)	Description
OCFJustWorks	oic.sec.doxm.jw	0	The just-works method relies on anonymous Diffie-Hellman key agreement protocol to allow a DOTS to assert ownership of the new Device. The first DOTS to make the assertion is accepted as the Device owner. The just-works method results in a shared secret that is used to authenticate the Device to the DOTS and likewise authenticates the DOTS to the Device. The Device permits the DOTS to take ownership of the Device, after which a second attempt to take ownership by a different DOTS will fail <sup>a</sup> .
OCFSharedPin	oic.sec.doxm.rdp	1	The new Device randomly generates a PIN that is communicated via an Out Of Band Communication Channel to a DOTS. An in-band Diffie-Hellman key agreement protocol establishes that both endpoints possess the PIN. Possession of the PIN by the DOTS signals the new Device that device ownership can be asserted.
OCFMfgCert	oic.sec.doxm.mfgcert	2	The new Device is presumed to have been manufactured with an embedded asymmetric private key that is used to sign a Diffie-Hellman exchange at Device onboarding. The manufacturer certificate should contain Platform hardening information and other security assurances assertions.
OCF Reserved	<Reserved>	3	Reserved
OCFSelf	oic.sec.doxm.self	4	The manufacturer shall set the "/doxm.doxmsel" value to (4). The Server shall reset this value to (4) upon entering RESET Device state.
OCF Reserved	<Reserved>	5~0xFEFF	Reserved for OCF use
Vendor-defined Value Type Name	<Reserved>	0xFF00~0xFFFF	Reserved for vendor-specific OTM use
<p><sup>a</sup> The just-works method is subject to a man-in-the-middle attacker. Precautions should be taken to provide physical security when this method is used.</p>			

2620 **13.3 Credential Resource**

2621 **13.3.1 Credential Resource General**

2622 The "/oic/sec/cred" Resource maintains credentials used to authenticate the Server to Clients and  
 2623 support services as well as credentials used to verify Clients and support services.

2624 Multiple credential types are anticipated by the OCF framework, including pair-wise pre-shared  
 2625 keys, asymmetric keys, certificates and others. The credential Resource uses a Subject UUID to  
 2626 distinguish the Clients and support services it recognizes by verifying an authentication challenge.

2627 In order to provide an interface which allows management of the "creds" Array Property, the  
 2628 RETRIEVE, UPDATE and DELETE operations on the "/oic/sec/cred" Resource shall behave as  
 2629 follows:

- 2630 1) A RETRIEVE shall return the full Resource representation, except that any write-only Properties  
 2631 shall be omitted (e.g. private key data).
- 2632 2) An UPDATE shall replace or add to the Properties included in the representation sent with the  
 2633 UPDATE request, as follows:
  - 2634 a) If an UPDATE representation includes the "creds" array Property, then:
    - 2635 i) Supplied "creds" with a "credid" that matches an existing "credid" shall replace  
 2636 completely the corresponding "cred" in the existing "creds" array.
    - 2637 ii) Supplied "creds" without a "credid" shall be appended to the existing "creds" array, and  
 2638 a unique (to the "cred" Resource) "credid" shall be created and assigned to the new  
 2639 "cred" by the Server. The "credid" of a deleted "cred" should not be reused, to improve  
 2640 the determinism of the interface and reduce opportunity for race conditions.
    - 2641 iii) Supplied "creds" with a "credid" that does not match an existing "credid" shall be  
 2642 appended to the existing "creds" array, using the supplied "credid".
    - 2643 iv) The rows in Table 20 corresponding to the "creds" array Property dictate the Device  
 2644 States in which an UPDATE of the "creds" array Property is always rejected. If OCF  
 2645 Device is in a Device State where the Access Mode in this row contains "R", then the  
 2646 OCF Device shall reject all UPDATES of the "creds" array Property.
- 2647 3) A DELETE without query parameters shall remove the entire "creds" array, but shall not remove  
 2648 the "/oic/sec/cred" Resource.
- 2649 4) A DELETE with one or more "credid" query parameters shall remove the "cred"(s) with the  
 2650 corresponding "credid"(s) from the "creds" array.
- 2651 5) The rows in Table 20 corresponding to the "creds" array Property dictate the Device States in  
 2652 which a DELETE is always rejected. If OCF Device is in a Device State where the Access Mode  
 2653 in this row contains "R", then the OCF Device shall reject all DELETES.

2654 NOTE The "/oic/sec/cred" Resource's use of the DELETE operation is not in accordance with the OCF Interfaces defined  
 2655 in ISO/IEC 30118-1:2018.

2656 "/oic/sec/cred" Resource is defined in Table 19.

2657 **Table 19 – Definition of the "/oic /sec/cred" Resource**

Fixed URI	Resource Type Title	Resource Type ID ("rt" value)	OCF Interfaces	Description	Related Functional Interaction
/oic/sec/cred	Credentials	oic.r.cred	baseline	Resource containing credentials for Device authentication, verification and data protection	Security

2658 Table 20 defines the Properties of the "/oic/sec/cred" Resource.

**Table 20 – Properties of the "/oic/sec/cred" Resource**

Property Title	Property Name	Value Type	Value Rule	Mandatory	Device State	Access Mode	Description
Credentials	creds	oic.sec.cred	array	Yes	RESET	R	Server shall set to manufacturer defaults.
					RFOTM	RW	Set by DOTS after successful OTM
					RFPRO	RW	Set by the CMS (referenced via the rowneruuid Property of "/oic/sec/cred" Resource) after successful authentication. Access to NCRs is prohibited.
					RFNOP	R	Access to NCRs is permitted after a matching ACE is found.
					SRESET	RW	The DOTS (referenced via devowneruuid Property of "/oic/sec/doxm" Resource or the rowneruuid Property of "/oic/sec/doxm" Resource) should evaluate the integrity of and may update creds entries when a secure session is established and the Server and DOTS are authenticated.
Resource Owner ID	rowneruuid	String	uuid	Yes	RESET	R	Server shall set to the nil uuid value (e.g. "00000000-0000-0000-0000-000000000000" )
					RFOTM	RW	The DOTS shall configure the rowneruuid Property of "/oic/sec/cred" Resource when a successful owner transfer session is established.
					RFPRO	R	n/a
					RFNOP	R	n/a
					SRESET	RW	The DOTS (referenced via devowneruuid Property of "/oic/sec/doxm" Resource or the rowneruuid Property of "/oic/sec/doxm" Resource) should verify and if needed, update the resource owner Property when a mutually authenticated secure session is established. If the "rowneruuid" Property does not refer to a valid DOTS the Server shall transition to RESET Device state.

2660 All secure Device accesses shall have a "/oic/sec/cred" Resource that protects the end-to-end  
2661 interaction.

2662 The "/oic/sec/cred" Resource shall be updateable by the service named in its rowneruuid Property.

2663 ACLs naming "/oic/sec/cred" Resource should further restrict access beyond CRUDN access  
2664 modes.

2665 Table 21 defines the Properties of "oic.sec.creds".

Table 21 – Properties of the "oic.sec.creds" Property

Property Title	Property Name	Value Type	Value Rule	Mandatory	Access Mode	Device State	Description
Credential ID	credid	UINT16	0 – 64K-1	Yes	RW		Short credential ID for local references from other Resource
Subject UUID	subjectuuid	String	uuid	Yes	RW		A uuid that identifies the subject to which this credential applies or "*" if any identity is acceptable
Role ID	roleid	oic.sec.roletype	-	No	RW		Identifies the role(s) the subject is authorized to assert.
Credential Type	credtype	oic.sec.credtype	bitmask	Yes	RW		Represents this credential's type. 0 – Used for testing 1 – Symmetric pair-wise key 2 – Symmetric group key 4 – Asymmetric signing key 8 – Asymmetric signing key with certificate 16 – PIN or password 32 – Asymmetric encryption key
Credential Usage	credusage	oic.sec.credusage	String	No	RW		Used to resolve undecidability of the credential. Provides indication for how/where the cred is used "oic.sec.cred.trustca": certificate trust anchor "oic.sec.cred.cert": identity certificate "oic.sec.cred.rolecert": role certificate "oic.sec.cred.mfgtrustca": manufacturer certificate trust anchor "oic.sec.cred.mfgcert": manufacturer certificate
Public Data	publicdata	oic.sec.pubdatatype	-	No	RW		Public credential information 1:2: ticket, public SKDC values 4, 32: Public key value 8: A chain of one or more certificate
Private Data	privatedata	oic.sec.privdatatype	-	No	-	RESET	Server shall set to manufacturer default
					RW	RFOTM	Set by DOTS after successful OTM
					W	RFPRO	Set by authenticated DOTS or CMS
					-	RFNOP	Not writable during normal operation.
					W	SRESET	DOTS may modify to enable transition to RFPRO.
Optional Data	optionaldata	oic.sec.optdatatype	-	No	RW		Credential revocation status information 1, 2, 4, 32: revocation status information 8: Revocation information
Period	period	String	-	No	RW		Period as defined by IETF RFC 5545. The credential should not be used if the current time is outside the Period window.
Credential Refresh Method	crms	oic.sec.crmtype	array	No	RW		Credentials with a Period Property are refreshed using the credential refresh method (crm) according to the type definitions for "oic.sec.crm".

2667 Table 22 defines the Properties of "oic.sec.credusagetype".

2668 **Table 22: Properties of the "oic.sec.credusagetype" Property**

Value Type Name	Value Type URN (mandatory)
Trust Anchor	oic.sec.cred.trustca
Certificate	oic.sec.cred.cert
Role Certificate	oic.sec.cred.rolecert
Manufacturer Trust CA	oic.sec.cred.mfgtrustca
Manufacturer CA	oic.sec.cred.mfgcert

2669 Table 23 defines the Properties of "oic.sec.pubdatatype".

2670 **Table 23 – Properties of the "oic.sec.pubdatatype" Property**

Property Title	Property Name	Value Type	Value Rule	Access Mode	Mandatory	Description
Encoding format	encoding	String	N/A	RW	No	A string specifying the encoding format of the data contained in the pubdata "oic.sec.encoding.pem" – Encoding for PEM-encoded certificate or chain
Data	data	String	N/A	RW	No	The encoded value

2671 Table 24 defines the Properties of "oic.sec.privdatatype".

2672 **Table 24 – Properties of the "oic.sec.privdatatype" Property**

Property Title	Property Name	Value Type	Value Rule	Access Mode	Mandatory	Description
Encoding format	encoding	String	N/A	RW	Yes	A string specifying the encoding format of the data contained in the privdata "oic.sec.encoding.pem" – Encoding for PEM-encoded private key "oic.sec.encoding.base64" – Encoding of Base64 encoded PSK "oic.sec.encoding.handle" – Data is contained in a storage sub-system referenced using a handle "oic.sec.encoding.raw" – Raw hex encoded data
Data	data	String	N/A	W	No	The encoded value This value shall not be RETRIEVE-able.
Handle	handle	UINT16	N/A	RW	No	Handle to a key storage resource

2673 Table 25 defines the Properties of "oic.sec.optdatatype".

**Table 25 – Properties of the "oic.sec.optdatatype" Property**

Property Title	Property Name	Value Type	Value Rule	Access Mode	Mandatory	Description
Revocation status	revstat	Boolean	T   F	RW	Yes	Revocation status flag True – revoked False – not revoked
Encoding format	encoding	String	N/A	RW	No	A string specifying the encoding format of the data contained in the optdata "oic.sec.encoding.pem" – Encoding for PEM-encoded certificate or chain
Data	data	String	N/A	RW	No	The encoded structure

2675 Table 26 defines the Properties of "oic.sec.roletype".

**Table 26 – Definition of the "oic.sec.roletype" type.**

Property Title	Property Name	Value Type	Value Rule	Access Mode	Mandatory	Description
Authority	authority	String	N/A	R	No	A name for the authority that defined the role. If not present, the credential issuer defined the role. If present, must be expressible as an ASN.1 PrintableString.
Role	role	String	N/A -	R	Yes	An identifier for the role. Must be expressible as an ASN.1 PrintableString.

### 2677 13.3.2 Properties of the Credential Resource

#### 2678 13.3.2.1 Credential ID

2679 Credential ID ("credid") is a local reference to an entry in a "creds" Property array of the  
2680 "/oic/sec/cred" Resource. The SRM generates it. The "credid" Property shall be used to  
2681 disambiguate array elements of the "creds" Property.

#### 2682 13.3.2.2 Subject UUID

2683 The "subjectuuid" Property identifies the Device to which an entry in a "creds" Property array of the  
2684 "/oic/sec/cred" Resource shall be used to establish a secure session, verify an authentication  
2685 challenge-response or to authenticate an authentication challenge.

2686 A "subjectuuid" Property that matches the Server's own "deviceuuid" Property, distinguishes the  
2687 array entries in the "creds" Property that pertain to this Device.

2688 The "subjectuuid" Property shall be used to identify a group to which a group key is used to protect  
2689 shared data.

2690 When certificate chain is used during secure connection establishment, the "subjectuuid" Property  
2691 shall also be used to verify the identity of the responder. The presented certificate chain shall be  
2692 accepted, if there is a matching Credential entry on the Device that satisfies all of the following:

- 2693 – Public Data of the entry contains trust anchor (root) of the presented chain.
- 2694 – Subject UUID of the entry matches UUID in the Common Name field of the End-Entity certificate  
2695 in the presented chain. If Subject UUID of the entry is set as a wildcard "\*", this condition is  
2696 automatically satisfied.
- 2697 – Credential Usage of the entry is "oic.sec.cred.trustca".



2698 **13.3.2.3 Role ID**

2699 The "roleid" Property identifies a role that has been granted to the credential.

2700 **13.3.2.4 Credential Type**

2701 The "credtype" Property is used to interpret several of the other Property values whose contents  
2702 can differ depending on credential type. These Properties include "publicdata", "privatedata" and  
2703 "optionaldata". The "credtype" Property value of "0" ("no security mode") is reserved for testing and  
2704 debugging circumstances. Production deployments shall not allow provisioning of credentials of  
2705 type "0". The SRM should introduce checking code that prevents its use in production deployments.

2706 **13.3.2.5 Public Data**

2707 The "publicdata" Property contains information that provides additional context surrounding the  
2708 issuance of the credential. For example, it might contain information included in a certificate or  
2709 response data from a CMS. It might contain wrapped data.

2710 **13.3.2.6 Private Data**

2711 The "privatedata" Property contains secret information that is used to authenticate a Device, protect  
2712 data or verify an authentication challenge-response.

2713 The "privatedata" Property shall not be disclosed outside of the SRM's trusted computing perimeter.  
2714 A secure element (SE) or trusted execution environment (TEE) should be used to implement the  
2715 SRM's trusted computing perimeter. The privatedata contents may be referenced using a handle;  
2716 for example, if used with a secure storage sub-system.

2717 **13.3.2.7 Optional Data**

2718 The "optionaldata" Property contains information that is optionally supplied, but facilitates key  
2719 management, scalability or performance optimization.

2720 **13.3.2.8 Period**

2721 The "period" Property identifies the validity period for the credential. If no validity period is specified,  
2722 the credential lifetime is undetermined. Constrained devices that do not implement a date-time  
2723 capability shall obtain current date-time information from its CMS.

2724 **13.3.2.9 Credential Refresh Method Type Definition [Deprecated]**

2725 This clause is intentionally left blank.

2726 **13.3.2.10 Credential Usage**

2727 Credential Usage indicates to the Device the circumstances in which a credential should be used.  
2728 Five values are defined:

- 2729 – "oic.sec.cred.trustca": This certificate is a trust anchor for the purposes of certificate chain  
2730 validation, as defined in 10.4. OCF Server SHALL remove any "/oic/sec/cred" entries with an  
2731 "oic.sec.cred.trustca" credusage upon transitioning to RFOTM. OCF Servers SHALL use  
2732 "/oic/sec/cred" entries that have an "oic.sec.cred.trustca" Value of "credusage" Property only  
2733 as trust anchors for post-onboarding (D)TLS session establishment in RFNOP state; these  
2734 entries are not to be used for onboarding (D)TLS sessions.
- 2735 – "oic.sec.cred.cert": This "credusage" is used for certificates for which the Device possesses the  
2736 private key and uses it for identity authentication in a secure session, as defined in clause 10.4.
- 2737 – "oic.sec.cred.rolecert": This "credusage" is used for certificates for which the Device possesses  
2738 the private key and uses to assert one or more roles, as defined in clause 10.4.2.
- 2739 – "oic.sec.cred.mfgtrustca": This certificate is a trust anchor for the purposes of the Manufacturer  
2740 Certificate Based OTM as defined in clause 7.3.6. OCF Servers SHALL use "/oic/sec/cred"

2741 entries that have an "oic.sec.cred.mfgtrustca" Value of "credusage" Property only as trust  
2742 anchors for onboarding (D)TLS session establishment; these entries are not to be used for post-  
2743 onboarding (D)TLS sessions.

2744 – "oic.sec.cred.mfgcert": This certificate is used for certificates for which the Device possesses  
2745 the private key and uses it for authentication in the Manufacturer Certificate Based OTM as  
2746 defined in clause 7.3.6.

### 2747 13.3.2.11 Resource Owner

2748 The Resource Owner Property allows credential provisioning to occur soon after Device onboarding  
2749 before access to support services has been established. It identifies the entity authorized to  
2750 manage the "/oic/sec/cred" Resource in response to Device recovery situations.

## 2751 13.3.3 Key Formatting

### 2752 13.3.3.1 Symmetric Key Formatting

2753 Symmetric keys shall have the format described in Table 27 and Table 28.

2754 **Table 27 – 128-bit symmetric key**

Name	Value	Type	Description
Length	16	OCTET	Specifies the number of 8-bit octets following Length
Key	opaque	OCTET Array	16-byte array of octets. When used as input to a PSK function Length is omitted.

2755

2756 **Table 28 – 256-bit symmetric key**

Name	Value	Type	Description
Length	32	OCTET	Specifies the number of 8-bit octets following Length
Key	opaque	OCTET Array	32-byte array of octets. When used as input to a PSK function Length is omitted.

### 2757 13.3.3.2 Asymmetric Keys

2758 Asymmetric key formatting is not available in this revision of the document.

### 2759 13.3.3.3 Asymmetric Keys with Certificate

2760 Key formatting is defined by certificate definition.

### 2761 13.3.3.4 Passwords

2762 Password formatting is not available in this revision of the document.

## 2763 13.3.4 Credential Refresh Method Details [Deprecated]

2764 This clause is intentionally left blank.

## 2765 13.4 Certificate Revocation List

### 2766 13.4.1 CRL Resource Definition [Deprecated]

2767 This clause is intentionally left blank.

## 2768 13.5 ACL Resources

### 2769 13.5.1 ACL Resources General

2770 All Resource hosted by a Server are required to match an ACL policy. ACL policies can be  
2771 expressed using "/oic/sec/acl2". The subject (e.g. "deviceuid" of the Client) requesting access to

2772 a Resource shall be authenticated prior to applying the ACL check. Resources that are available  
 2773 to multiple Clients can be matched using a wildcard subject. All Resources accessible via the  
 2774 unsecured communication endpoint shall be matched using a wildcard subject.

2775 **13.5.2 OCF Access Control List (ACL) BNF defines ACL structures.**

2776 ACL structure in Backus-Naur Form (BNF) notation is defined in Table 29:

2777 **Table 29 – BNF Definition of OCF ACL**

<ACL>	<ACE> {<ACE>}
<ACE>	<SubjectId> <ResourceRef> <Permission> {<Validity>}
<SubjectId>	<DeviceId>   <Wildcard>   <RoleId>
<DeviceId>	<UUID>
<RoleId>	<Character>   <RoleName><Character>
<RoleName>	" "   <Authority><Character>
<Authority>	<UUID>
<ResourceRef>	' (' <OIC_LINK> {',' {OIC_LINK}} ')'
<Permission>	('C'   '-' ) ('R'   '-' ) ('U'   '-' ) ('D'   '-' ) ('N'   '-' )
<Validity>	<Period> {<Recurrence>}
<Wildcard>	'*'
<URI>	IETF RFC 3986
<UUID>	IETF RFC 4122
<Period>	IETF RFC 5545 Period
<Recurrence>	IETF RFC 5545 Recurrence
<OIC_LINK>	ISO/IEC 30118-1:2018 defined in JSON Schema
<Character>	<Any UTF8 printable character, excluding NUL>

2778 The <DeviceId> token means the requestor must possess a credential that uses <UUID> as its  
 2779 identity in order to match the requestor to the <ACE> policy.

2780 The <RoleId> token means the requestor must possess a role credential with <Character> as its  
 2781 role in order to match the requestor to the <ACE> policy.

2782 The <Wildcard> token "\*" means any requestor is matched to the <ACE> policy, with or without  
 2783 authentication.

2784 When a <SubjectId> is matched to an <ACE> policy the <ResourceRef> is used to match the <ACE>  
 2785 policy to Resources.

2786 The <OIC\_LINK> token contains values used to query existence of hosted Resources.

2787 The <Permission> token specifies the privilege granted by the <ACE> policy given the <SubjectId>  
 2788 and <ResourceRef> matching does not produce the empty set match.

2789 Permissions are defined in terms of CREATE ("C"), RETRIEVE ("R"), UPDATE ("U"), DELETE ("D"),  
 2790 NOTIFY ("N") and NIL ("-"). NIL is substituted for a permissions character that signifies the  
 2791 respective permission is not granted.

2792 The empty set match result defaults to a condition where no access rights are granted.

2793 If the <Validity> token exists, the <Permission> granted is constrained to the time <Period>.  
 2794 <Validity> may further be segmented into a <Recurrence> pattern where access may alternatively  
 2795 be granted and rescinded according to the pattern.

2796 **13.5.3 ACL Resource**

2797 An "acl2" is a list of type "ace2".

2798 In order to provide an interface which allows management of array elements of the "aclist2"  
2799 Property associated with a "/oic/sec/acl2" Resource. The RETRIEVE, UPDATE and DELETE  
2800 operations on the " /oic/sec/acl2" Resource SHALL behave as follows:

- 2801 1) A RETRIEVE shall return the full Resource representation.
- 2802 2) An UPDATE shall replace or add to the Properties included in the representation sent with the  
2803 UPDATE request, as follows:
- 2804 a) If an UPDATE representation includes the array Property, then:
- 2805 i) Supplied ACEs with an "aceid" that matches an existing "aceid" shall replace completely  
2806 the corresponding ACE in the existing "aces2" array.
- 2807 ii) Supplied ACEs without an "aceid" shall be appended to the existing "aces2" array, and  
2808 a unique (to the acl2 Resource) "aceid" shall be created and assigned to the new ACE  
2809 by the Server. The "aceid" of a deleted ACE should not be reused, to improve the  
2810 determinism of the interface and reduce opportunity for race conditions.
- 2811 iii) Supplied ACEs with an "aceid" that does not match an existing "aceid" shall be  
2812 appended to the existing "aces2" array, using the supplied "aceid".
- 2813 iv) The rows in Table 32 corresponding to the "aclist2" array Property dictate the Device  
2814 States in which an UPDATE of the "aclist2" array Property is always rejected. If OCF  
2815 Device is in a Device State where the Access Mode in this row contains "R", then the  
2816 OCF Device shall reject all UPDATES of the "aclist2" array Property.
- 2817 3) A DELETE without query parameters shall remove the entire "aces2" array, but shall not remove  
2818 the "oic/sec/ace2" Resource.
- 2819 4) A DELETE with one or more "aceid" query parameters shall remove the ACE(s) with the  
2820 corresponding "aceid"(s) from the "aces2" array.
- 2821 5) The rows in Table 32 corresponding to the "aclist2" array Property dictate the Device States in  
2822 which a DELETE is always rejected. If OCF Device is in a Device State where the Access Mode  
2823 in this row contains "R", then the OCF Device shall reject all DELETES.

2824 NOTE The "/oic/sec/acl2" Resource's use of the DELETE operation is not in accordance with the OCF Interfaces  
2825 defined in ISO/IEC 30118-1:2018.

2826 Evaluation of local ACL Resource completes when all ACL Resource have been queried and no  
2827 entry can be found for the requested Resource for the requestor – e.g. "/oic/sec/acl2" does not  
2828 match the subject and the requested Resource.

2829 Table 30 defines the values of "oic.sec.crudntype".

2830

**Table 30 – Value Definition of the "oic.sec.crudntype" Property**

Value	Access Policy	Description	RemarksNotes
bx0000,0000 (0)	No permissions	No permissions	N/A
bx0000,0001 (1)	C	CREATE	N/A
bx0000,0010 (2)	R	RETRIEVE, OBSERVE, DISCOVER	The "R" permission bit covers both the Read permission and the Observe permission.
bx0000,0100 (4)	U	WRITE, UPDATE	N/A
bx0000,1000 (8)	D	DELETE	N/A
bx0001,0000 (16)	N	NOTIFY	The "N" permission bit is ignored in OCF 1.0, since "R" covers the Observe permission. It is documented for future versions

2831 "oic/sec/acl2" Resource is defined in Table 19.

2832

**Table 31 – Definition of the "oic/sec/acl2" Resource**

Fixed URI	Resource Type Title	Resource Type ID ("rt" value)	OCF Interfaces	Description	Related Functional Interaction
/oic/sec/acl2	ACL2	oic.r.acl2	baseline	Resource for managing access	Security

2833 Table 32 defines the Properties of "oic.sec.acl2".

**Table 32 – Properties of the "/oic/sec/acl2" Resource**

Property Name	Value Type	Mandatory	Device State	Access Mode	Description
aclist2	array of oic.sec.ace2	Yes	N/A		The aclist2 Property is an array of ACE records of type "oic.sec.ace2". The Server uses this list to apply access control to its local resources.
N/A	N/A	N/A	RESET	R	Server shall set to manufacturer defaults.
			RFOTM	RW	Set by DOTS after successful OTM
			RFPRO	RW	The AMS (referenced via rowneruuid property) shall update the aclist entries after mutually authenticated secure session is established. Access to NCRs is prohibited.
			RFNOP	R	Access to NCRs is permitted after a matching ACE2 is found.
			SRESET	RW	The DOTS (referenced via devowneruuid Property of "/oic/sec/doxm Resource") should evaluate the integrity of and may update aclist entries when a secure session is established and the Server and DOTS are authenticated.
rowneruuid	uuid	Yes	N/A		The resource owner Property (rowneruuid) is used by the Server to reference a service provider trusted by the Server. Server shall verify the service provider is authorized to perform the requested action
			RESET	R	Server shall set to the nil uuid value (e.g. "00000000-0000-0000-0000-000000000000" )
			RFOTM	RW	The DOTS should configure the rowneruuid Property of "/oic/sec/acl2" Resource when a successful owner transfer session is established.
			RFPRO	R	n/a
			RFNOP	R	n/a
			SRESET	RW	The DOTS (referenced via devowneruuid Property or rowneruuid Property of "/oic/sec/doxm" Resource) should verify and if needed, update the resource owner Property when a mutually authenticated secure session is established. If the rowneruuid Property does not refer to a valid DOTS the Server shall transition to RESET device state.

2835

2836 Table 33 defines the Properties of "oic.sec.ace2".

2837

**Table 33 – "oic.sec.ace2" data type definition.**

Property Name	Value Type	Mandatory	Description
subject	oic.sec.roletype, oic.sec.didtype, oic.sec.conntype	Yes	The Client is the subject of the ACE when the roles, Device ID, or connection type matches.
resources	array of oic.sec.ace2.resource -ref	Yes	The application's resources to which a security policy applies
permission	oic.sec.crudntype.bitmask	Yes	Bitmask encoding of CRUDN permission
validity	array of oic.sec.time-pattern	No	An array of a tuple of period and recurrence. Each item in this array contains a string representing a period using the IETF RFC 5545 Period, and a string array representing a recurrence rule using the IETF RFC 5545 Recurrence.
aceid	integer	Yes	An aceid is unique with respect to the array entries in the aclist2 Property.

2838 Table 34 defines the Properties of "oic.sec.ace2.resource-ref".

2839

**Table 34 – "oic.sec.ace2.resource-ref" data type definition.**

Property Name	Value Type	Mandatory	Description
href	uri	No	A URI referring to a resource to which the containing ACE applies
wc	string	No	Refer to Table 14.

2840 Table 35 defines the values of "oic.sec.ace2.resource-ref".

2841

**Table 35 – Value definition "oic.sec.conntype" Property**

Property Name	Value Type	Value Rule	Description
conntype	string	enum [ "auth-crypt", "anon-clear" ]	This Property allows an ACE to be matched based on the connection or message protection type
		auth-crypt	ACE applies if the Client is authenticated and the data channel or message is encrypted and integrity protected
		anon-clear	ACE applies if the Client is not authenticated and the data channel or message is not encrypted but may be integrity protected

2842 Local ACL Resources supply policy to a Resource access enforcement point within an OCF stack  
 2843 instance. The OCF framework gates Client access to Server Resources. It evaluates the subject's  
 2844 request using policies contained in ACL resources.

2845 Resources named in the ACL policy can be fully qualified or partially qualified. Fully qualified  
 2846 Resource references include the device identifier in the href Property that identifies the remote  
 2847 Resource Server that hosts the Resource. Partially qualified references mean that the local  
 2848 Resource Server hosts the Resource. If a fully qualified resource reference is given, the  
 2849 Intermediary enforcing access shall have a secure channel to the Resource Server and the  
 2850 Resource Server shall verify the Intermediary is authorized to act on its behalf as a Resource  
 2851 access enforcement point.

2852 Resource Servers should include references to Device and ACL Resources where access  
2853 enforcement is to be applied. However, access enforcement logic shall not depend on these  
2854 references for access control processing as access to Server Resources will have already been  
2855 granted.

2856 Local ACL Resources identify a Resource Owner service that is authorized to instantiate and modify  
2857 this Resource. This prevents non-terminating dependency on some other ACL Resource.  
2858 Nevertheless, it should be desirable to grant access rights to ACL Resources using an ACL  
2859 Resource.

2860 An ACE2 entry is considered "currently valid" if the validity period of the ACE2 entry includes the  
2861 time of the request. The validity period in the ACE2 may be a recurring time period (e.g., daily from  
2862 1:00-2:00). Matching the resource(s) specified in a request to the "resource" Property of the ACE2  
2863 is defined in clause 12.2. For example, one way they can match is if the Resource URI in the  
2864 request exactly matches one of the resource references in the ACE2 entries.

2865 A request will match an ACE2 if any of the following are true:

2866 1) The ACE2 "subject" Property is of type "oic.sec.didtype" has a UUID value that matches the  
2867 "deviceuuid" Property associated with the secure session;

2868 AND the Resource of the request matches one of the "resources" Property of the ACE2  
2869 "oic.sec.ace2.resource-ref";

2870 AND the ACE2 is currently valid.

2871 2) The ACE2 "subject" Property is of type "oic.sec.conntype" and has the wildcard value that  
2872 matches the currently established connection type;

2873 AND the resource of the request matches one of the "resources" Property of the ACE2  
2874 "oic.sec.ace2.resource-ref";

2875 AND the ACE2 is currently valid.

2876 3) When Client authentication uses a certificate credential;

2877 AND one of the "roleid" values contained in the role certificate matches the "roleid" Property of  
2878 the ACE2 "oic.sec.roletype";

2879 AND the role certificate public key matches the public key of the certificate used to establish  
2880 the current secure session;

2881 AND the resource of the request matches one of the array elements of the "resources" Property  
2882 of the ACE2 "oic.sec.ace2.resource-ref";

2883 AND the ACE2 is currently valid.

2884 4) When Client authentication uses a certificate credential;

2885 AND the CoAP payload query string of the request specifies a role, which is member of the set  
2886 of roles contained in the role certificate;

2887 AND the roleid values contained in the role certificate matches the "roleid" Property of the ACE2  
2888 "oic.sec.roletype";

2889 AND the role certificate public key matches the public key of the certificate used to establish  
2890 the current secure session;

2891 AND the resource of the request matches one of the "resources" Property of the ACE2  
2892 "oic.sec.ace2.resource-ref";

2893 AND the ACE2 is currently valid.

2894 5) When Client authentication uses a symmetric key credential;

2895 AND one of the "roleid" values associated with the symmetric key credential used in the secure  
2896 session, matches the "roleid" Property of the ACE2 "oic.sec.roletype";



2897 AND the resource of the request matches one of the array elements of the "resources" Property  
2898 of the ACE2 "oic.sec.ace2.resource-ref";

2899 AND the ACE2 is currently valid.

2900 6) When Client authentication uses a symmetric key credential;

2901 AND the CoAP payload query string of the request specifies a role, which is contained in the  
2902 "oic.r.cred.creds.roleid" Property of the current secure session;

2903 AND CoAP payload query string of the request specifies a role that matches the "roleid"  
2904 Property of the ACE2 "oic.sec.roletype";

2905 AND the resource of the request matches one of the array elements of the "resources" Property  
2906 of the ACE2 "oic.sec.ace2.resource-ref";

2907 AND the ACE2 is currently valid.

2908 A request is granted if ANY of the 'matching' ACE2 entries contain the permission to allow the  
2909 request. Otherwise, the request is denied.

2910 There is no way for an ACE2 entry to explicitly deny permission to a resource. Therefore, if one  
2911 Device with a given role should have slightly different permissions than another Device with the  
2912 same role, they must be provisioned with different roles.

2913 The Server is required to verify that any hosted Resource has authorized access by the Client  
2914 requesting access. The "/oic/sec/acl2" Resource is co-located on the Resource host so that the  
2915 Resource request processing should be applied securely and efficiently. See Annex A for example.

### 2916 **13.6 Access Manager ACL Resource [Deprecated]**

2917 This clause is intentionally left blank.

### 2918 **13.7 Signed ACL Resource [Deprecated]**

2919 This clause is intentionally left blank.

### 2920 **13.8 Provisioning Status Resource**

2921 The "/oic/sec/pstat" Resource maintains the Device provisioning status. Device provisioning should  
2922 be Client-directed or Server-directed. Client-directed provisioning relies on a Client device to  
2923 determine what, how and when Server Resources should be instantiated and updated. Server-  
2924 directed provisioning relies on the Server to seek provisioning when conditions dictate. Furthermore,  
2925 the "/oic/sec/cred" Resource should be provisioned at ownership transfer with credentials  
2926 necessary to open a secure connection with appropriate support service.

2927 "/oic/sec/pstat" Resource is defined in Table 36.

2928 **Table 36 – Definition of the "/oic/sec/pstat" Resource**

Fixed URI	Resource Type Title	Resource Type ID ("rt" value)	OCF Interfaces	Description	Related Functional Interaction
/oic/sec/pstat	Provisioning Status	oic.r.pstat	baseline	Resource for managing Device provisioning status	Configuration

2929 Table 37 defines the Properties of "/oic/sec/pstat".

Table 37 – Properties of the "/oic/sec/pstat" Resource

Property Title	Property Name	Value Type	Value Rule	Mandatory	Access Mode	Device State	Description
Device Onboarding State	dos	oic.sec.dostype	N/A	Yes	RW		Device Onboarding State
Is Device Operational	isop	Boolean	T F	Yes	R	RESET	Server shall set to FALSE
					R	RFOTM	Server shall set to FALSE
					R	RFPRO	Server shall set to FALSE
					R	RFNOP	Server shall set to TRUE
					R	SRESET	Server shall set to FALSE
Current Mode	cm	oic.sec.dpmttype	bitmask	Yes	R		Current Mode
Target Mode	tm	oic.sec.dpmttype	bitmask	Yes	RW		Target Mode
Operational Mode	om	oic.sec.pomttype	bitmask	Yes	R	RESET	Server shall set to manufacturer default.
					RW	RFOTM	Set by DOTS after successful OTM
					RW	RFPRO	Set by CMS, AMS, DOTS after successful authentication
					RW	RFNOP	Set by CMS, AMS, DOTS after successful authentication
					RW	SRESET	Set by DOTS.
Supported Mode	sm	oic.sec.pomttype	bitmask	Yes	R	All states	Supported provisioning services operation modes
Device UUID	deviceuuid	String	uuid	Yes	RW	All states	[DEPRECATED] A uuid that identifies the Device to which the status applies
Resource Owner ID	rowneruid	String	uuid	Yes	R	RESET	Server shall set to the nil uuid value (e.g. "00000000-0000-0000-0000-000000000000" )
					RW	RFOTM	The DOTS should configure the rowneruid Property when a successful owner transfer session is established.
					R	RFPRO	n/a
					R	RFNOP	n/a
					RW	SRESET	The DOTS (referenced via devowneruid Property of "/oic/sec/doxm" Resource) should verify and if needed, update the resource owner Property when a mutually authenticated secure session is established. If the rowneruid does not refer to a valid DOTS the Server shall transition to RESET Device state.

**Table 38 – Properties of the ".oic.sec.dostype" Property**

Property Title	Property Name	Value Type	Value Rule	Mandator y	Access Mode	Device State	Description
Device Onboarding State	s	UINT16	enum (0=RESET, 1=RFOTM, 2=RFPRO, 3=RFNOP, 4=SRESET	Y	R	RESET	The Device is in a hard reset state.
					RW	RFOTM	Set by DOTS after successful OTM to RFPRO.
					RW	RFPRO	Set by CMS, AMS, DOTS after successful authentication
					RW	RFNOP	Set by CMS, AMS, DOTS after successful authentication
					RW	SRESET	Set by CMS, AMS, DOTS after successful authentication
Pending state	p	Boolean	T   F	Y	R	All States	FALSE (0) – "s" state changes are complete. Since Device is not able to respond when the value is TRUE, other values of this property are DEPRECATED.

2934 In all Device states:

- 2935 – The Device permits an authenticated and authorised Client to change the Device state of a  
2936 Device by updating the "s" Property of the "dos" Property of the "/oic/sec/pstat" Resource to  
2937 the desired value. The allowed Device state transitions are defined in Figure 18.
- 2938 – Prior to updating the "s" Property of the "dos" Property of the "/oic/sec/pstat" Resource, the  
2939 Client configures the Device to meet entry conditions for the new Device state. The SVR  
2940 definitions define the entity (Client or Server) expected to perform the specific SVR  
2941 configuration change to meet the entry conditions. Once the Client has configured the aspects  
2942 for which the Client is responsible, it can update the "s" Property of the "dos" Property of the  
2943 "/oic/sec/pstat" Resource. The Server then makes any changes for which the Server is  
2944 responsible, including updating required SVR values, and set the "s" Property of the "dos"  
2945 Property of the "/oic/sec/pstat" Resource to the new value.

2946 When Device state is RESET:

- 2947 – All SVR content is removed and reset to manufacturer default values.
- 2948 – The default manufacturer Device state is RESET.
- 2949 – NCRs are reset to manufacturer default values.
- 2950 – NCRs shall not be accessible.
- 2951 – After successfully processing RESET the SRM transitions to RFOTM by setting the "s" Property  
2952 of the "dos" Property of the "/oic/sec/pstat" Resource to 1 (RFOTM).

2953 When Device state is RFOTM:

- 2954 – NCRs shall not be accessible.
- 2955 – Before OTM is successful, the the "s" Property of the "dos" Property of the "/oic/sec/pstat"  
2956 Resource is read-only by unauthenticated requestors
- 2957 – After the OTM is successful, the "s" Property of the "dos" Property of the "/oic/sec/pstat"  
2958 Resource is read-write by authorized requestors.
- 2959 – The negotiated Device OC is used to create an authenticated session over which the DOTS  
2960 directs the Device state to transition to RFPRO.

- 2961 – If an authenticated session cannot be established the ownership transfer session should be  
2962 disconnected and SRM sets back the Device state to RESET state.
- 2963 – Ownership transfer session, especially Random PIN OTM, should not exceed 60 seconds. If  
2964 the SRM asserts the OTM failed, the ownership transfer session should be disconnected, and  
2965 the Device should transition to RESET ("/pstat.dos.s"=0 (RESET)).
- 2966 – The DOTS UPDATES the "devowneruuid" Property in the "/oic/sec/doxm" Resource to a non-  
2967 nil UUID value. The DOTS (or other authorized client) can update it multiple times while in  
2968 RFOTM. It is not updatable while in other device states except when the Device state returns  
2969 to RFOTM through RESET.
- 2970 – The DOTS can have additional provisioning tasks to perform while in RFOTM. When done, the  
2971 DOTS UPDATES the "owned" Property in the "/oic/sec/doxm" Resource to "true".
- 2972 – After successful OTM, the DOTS triggers the transition to RFPRO state and the "s" Property of  
2973 the "dos" Property of the "/oic/sec/pstat" Resource is set to 2 (RFPRO).
- 2974 When Device state is RFPRO:
- 2975 – The "s" Property of the "dos" Property of the "/oic/sec/pstat" Resource is read-only by  
2976 unauthorized requestors and read-write by authorized requestors.
- 2977 – NCRs shall not be accessible, except for Easy Setup Resources, if supported.
- 2978 – An authorized Client may provision SVRs as needed for normal functioning in RFNOP.
- 2979 – An authorized Client may perform consistency checks on SVRs to determine which shall be re-  
2980 provisioned.
- 2981 – Failure to successfully provision SVRs may trigger a state change to RESET. For example, if  
2982 the Device has already transitioned from SRESET but consistency checks continue to fail.
- 2983 – The authorized Client sets the "s" Property of the "dos" Property of the "/oic/sec/pstat" Resource  
2984 to 3 (RFNOP).
- 2985 When Device state is RFNOP:
- 2986 – The "s" Property of the "dos" Property of the "/oic/sec/pstat" Resource is read-only by  
2987 unauthorized requestors and read-write by authorized requestors.
- 2988 – NCRs, SVRs and core Resources are accessible following normal access processing.
- 2989 – When additional provisioning is necessary, the Device may be transitioned to RFPRO by an  
2990 authorized Client. Only the Device owner should transition to SRESET or RESET.
- 2991 When Device state is SRESET:
- 2992 – NCRs shall not be accessible. The integrity of NCRs may be suspect but the SRM doesn't  
2993 attempt to access or reference them.
- 2994 – SVR integrity is not guaranteed, but access to some SVR Properties is necessary. These  
2995 include "devowneruuid" Property of the "/oic/sec/doxm" Resource,  
2996 "creds":[{"subjectuuid":<devowneruuid>},...] Property of the "/oic/sec/cred" Resource and  
2997 "pstat.dos.s" "/oic/sec/pstat" Resource.
- 2998 – The certificates that identify and authorize the Device owner are sufficient to re-create  
2999 minimalist "/oic/sec/cred" and "/oic/sec/doxm" Resources enabling Device owner control of  
3000 SRESET. If the SRM can't establish these Resources, then it will transition to RESET state.
- 3001 – An authorized Client performs SVR consistency checks. The authorized Client can provision  
3002 SVRs as needed to ensure they are available for continued provisioning in RFPRO or for normal  
3003 functioning in RFNOP.
- 3004 – The authorized Device owner can avoid entering RESET state and RFOTM by UPDATING  
3005 "pstat.dos.s" with RFPRO or RFNOP values.

3006 – ACLs on SVR are presumed to be invalid. Access authorization is granted according to Device  
3007 owner privileges only.

3008 – The SRM asserts a Client-directed operational mode (e.g. "/pstat.om"=4).

3009 The provisioning mode type is a 16-bit mask enumerating the various Device provisioning modes.  
3010 "{ProvisioningMode}" should be used in this document to refer to an instance of a provisioning  
3011 mode without selecting any particular value.

3012 "oic.sec.dpmttype" is defined in Table 39.

3013 **Table 39 – Definition of the "oic.sec.dpmttype" Property**

Type Name	Type URN	Description
Device Provisioning Mode	oic.sec.dpmttype	Device provisioning mode is a 16-bit bitmask describing various provisioning modes

3014 Table 40 and Table 41 define the values of "oic.sec.dpmttype".

3015 **Table 40 – Value Definition of the "oic.sec.dpmttype" Property (Low-Byte)**

Value	Device Mode	Description
bx0000,0001 (1)	Deprecated	
bx0000,0010 (2)	Deprecated	
bx0000,0100 (4)	Deprecated	
bx0000,1000 (8)	Deprecated	
bx0001,0000 (16)	Deprecated	
bx0010,0000 (32)	Deprecated	
bx0100,0000 (64)	Initiate Software Version Validation	Software version validation requested/pending (1) Software version validation complete (0) Requires software download to verify integrity of software package
bx1000,0000 (128)	Initiate Secure Software Update	Secure software update requested/pending (1) Secure software update complete (0)

3016 **Table 41 – Value Definition of the "oic.sec.dpmttype" Property (High-Byte)**

Value	Device Mode	Description
bx0000,0001 (1)	Initiate Software Availability Check	Checks if new software is available on remote endpoint. Does not require to download software. Methods used are out of bound.
Bits 2-8	<Reserved>	Reserved for later use

3017 The provisioning operation mode type is an 8-bit mask enumerating the various provisioning  
3018 operation modes.

3019 "oic.sec.pomtype" is defined in Table 42.

3020 **Table 42 – Definition of the "oic.sec.pomtype" Property**

Type Name	Type URN	Description
Device Provisioning OperationMode	oic.sec.pomtype	Device provisioning operation mode is a 8-bit bitmask describing various provisioning operation modes

3021 Table 43 defines the values of "oic.sec.pomtype".

3022

**Table 43 – Value Definition of the "oic.sec.pomtype" Property**

Value	Operation Mode	Description
bx0000,0001 (1)	Server-directed utilizing multiple provisioning services	Deprecated
bx0000,0010 (2)	Server-directed utilizing a single provisioning service	Deprecated
bx0000,0100 (4)	Client-directed provisioning	Device supports provisioning service control of this Device's provisioning operations. This bit is always TRUE.
bx0000,1000(8) – bx1000,0000(128)	<Reserved>	Reserved for later use
bx1111,11xx	<Reserved>	Reserved for later use

3023 **13.9 Certificate Signing Request Resource**

3024 The "/oic/sec/csr" Resource is used by a Device to provide its desired identity, public key to be  
 3025 certified, and a proof of possession of the corresponding private key in the form of a IETF RFC  
 3026 2986 PKCS#10 Certification Request. If the Device supports certificates (i.e. the "sct" Property of  
 3027 "/oic/sec/doxm" Resource has a 1 in the 0x8 bit position), the Device shall have a "/oic/sec/csr"  
 3028 Resource.

3029 "/oic/sec/csr" Resource is defined in Table 44.

3030

**Table 44 – Definition of the "/oic/sec/csr" Resource**

Fixed URI	Resource Type Title	Resource Type ID ("rt" value)	OCF Interfaces	Description	Related Functional Interaction
/oic/sec/csr	Certificate Signing Request	oic.r.csr	baseline	The CSR resource contains a Certificate Signing Request for the Device's public key.	Configuration

3031 Table 45 defines the Properties of "/oic/sec/csr".

3032

**Table 45 – Properties of the "oic.r.csr" Resource**

Property Title	Property Name	Value Type	Access Mode	Mandatory	Description
Certificate Signing Request	csr	String	R	Yes	Contains the signed CSR encoded according to the encoding Property
Encoding	encoding	String	R	Yes	A string specifying the encoding format of the data contained in the csr Property "oic.sec.encoding.pem" – Encoding for PEM-encoded certificate signing request

3033 The Device chooses which public key to use, and may optionally generate a new key pair for this  
 3034 purpose.

3035 In the CSR, the Common Name component of the Subject Name shall contain a string of the format  
 3036 "uuid:X" where X is the Device's requested UUID in the format defined by IETF RFC 4122. The  
 3037 Common Name, and other components of the Subject Name, may contain other data. If the Device  
 3038 chooses to include additional information in the Common Name component, it shall delimit it from  
 3039 the UUID field by white space, a comma, or a semicolon.

3040 If the Device does not have a pre-provisioned key pair to use, but is capable and willing to generate  
3041 a new key pair, the Device may begin generation of a key pair as a result of a RETRIEVE of this  
3042 resource. If the Device cannot immediately respond to the RETRIEVE request due to time required  
3043 to generate a key pair, the Device shall return an "operation pending" error. This indicates to the  
3044 Client that the Device is not yet ready to respond, but will be able at a later time. The Client should  
3045 retry the request after a short delay.

### 3046 **13.10 Roles Resource**

3047 The "roles" Resource maintains roles that have been asserted with role certificates, as described  
3048 in clause 10.4.2. Asserted roles have an associated public key, i.e., the public key in the role  
3049 certificate. Servers shall only grant access to the roles information associated with the public key  
3050 of the Client. The roles Resource should be viewed as an extension of the (D)TLS session state.  
3051 See 10.4.2 for how role certificates are validated.

3052 The roles Resource shall be created by the Server upon establishment of a secure (D)TLS session  
3053 with a Client, if is not already created. The roles Resource shall only expose a secured OCF  
3054 Endpoint in the "/oic/res" response. A Server shall retain the roles Resource at least as long as the  
3055 (D)TLS session exists. A Server shall retain each certificate in the roles Resource at least until the  
3056 certificate expires or the (D)TLS session ends, whichever is sooner. The requirements of clause  
3057 10.3 and 10.4.2 to validate a certificate's time validity at the point of use always apply. A Server  
3058 should regularly inspect the contents of the roles resource and purge contents based on a policy it  
3059 determines based on its resource constraints. For example, expired certificates, and certificates  
3060 from Clients that have not been heard from for some arbitrary period of time could be candidates  
3061 for purging.

3062 The OCF namespace ("oic.role.\*") is restricted to OCF-defined roles. "oic.role.owner" is an OCF-  
3063 defined Role that is intended to provide Resource Owner privileges to multiple Clients in a scalable  
3064 way. Servers shall grant access to perform all supported operations in the current Device state  
3065 (see clause 8) on all supported SVRs regardless of ACL configuration the Clients asserting  
3066 "oic.role.owner" Role. Servers shall reject assertion of any Role, which starts with "oic.role.", but  
3067 is not one of the following Roles:

3068 – "oic.role.owner"

3069 The "roles" Resource is implicitly created by the Server upon establishment of a (D)TLS session.  
3070 In more detail, the RETRIEVE, UPDATE and DELETE operations on the roles Resource shall  
3071 behave as follows. Unlisted operations are implementation specific and not reliable.

- 3072 1) A RETRIEVE request shall return all previously asserted roles associated with the currently  
3073 connected and authenticated Client's identity. RETRIEVE requests with a "credid" query  
3074 parameter is not supported; all previously asserted roles associated with the currently  
3075 connected and authenticated Client's identity are returned.
- 3076 2) An UPDATE request that includes the "roles" Property shall replace or add to the Properties  
3077 included in the array as follows:
- 3078 a) If either the "publicdata" or the "optionaldata" are different than the existing entries in the  
3079 "roles" array, the entry shall be added to the "roles" array with a new, unique "credid" value.
- 3080 b) If both the "publicdata" and the "optionaldata" match an existing entry in the "roles" array,  
3081 the entry shall be considered to be the same. The Server shall reply with a 2.04 Changed  
3082 response and a duplicate entry shall not be added to the array.
- 3083 c) The "credid" Property is optional in an UPDATE request and if included, it may be ignored  
3084 by the Server. The Server shall assign a unique "credid" value for every entry of the "roles"  
3085 array.

3086 3) A DELETE request without a "credid" query parameter shall remove all entries from the  
 3087 "/oic/sec/roles" resource array corresponding to the currently connected and authenticated  
 3088 Client's identity.

3089 4) A DELETE request with a "credid" query parameter shall remove only the entries of the  
 3090 "/oic/sec/roles" resource array corresponding to the currently connected and authenticated  
 3091 Client's identity and where the corresponding "credid" matches the entry.

3092 NOTE The "/oic/sec/roles" Resource's use of the DELETE operation is not in accordance with the OCF Interfaces  
 3093 defined in ISO/IEC 30118-1:2018.

3094 See clause 8 for restrictions on the states in which this Resource may be modified.

3095 "/oic/sec/roles" Resource is defined in Table 46.

3096 **Table 46 – Definition of the "/oic/sec/roles" Resource**

Fixed URI	Resource Type Title	Resource Type ID ("rt" value)	OCF Interfaces	Description	Related Functional Interaction
/oic/sec/roles	Roles	oic.r.roles	baseline	Resource containing roles that have previously been asserted to this Server	Security

3097 Table 47 defines the Properties of "/oic/sec/roles".

3098 **Table 47 – Properties of the "/oic/sec/roles" Resource**

Property Title	Property Name	Value Type	Value Rule	Access Mode	Mandatory	Description
Roles	roles	oic.sec.cred	array	RW	Yes	List of roles previously asserted to this Server

3099 Because "/oic/sec/roles" shares the "oic.sec.cred" schema with "/oic/sec/cred", "subjectuid" is a required Property.  
 3100 However, "subjectuid" is not used in a role certificate. Therefore, a Device may ignore the "subjectuid" Property if the  
 3101 Property is contained in an UPDATE request to the "/oic/sec/roles" Resource.

3102 **13.11 Account Resource – moved to OCF Cloud Security document**

3103 This clause is intentionally left blank.

3104 **13.12 Account Session Resource – moved to OCF Cloud Security document**

3105 This clause is intentionally left blank.

3106 **13.13 Account Token Refresh Resource – moved to OCF Cloud Security document**

3107 This clause is intentionally left blank.

3108 **13.14 Security Virtual Resources (SVRs) and Access Policy**

3109 The SVRs expose the security-related Properties of the Device.

3110 Granting access requests (RETRIEVE, UPDATE, DELETE, etc.) for these SVRs to unauthenticated  
 3111 (anonymous) Clients could create privacy or security concerns.

3112 For example, when the Device onboarding State is RFOTM, it is necessary to grant requests for  
 3113 the "/oic/sec/doxm" Resource to anonymous requesters, so that the Device can be discovered and  
 3114 onboarded by an OBT. Subsequently, it might be preferable to deny requests for the  
 3115 "/oic/sec/doxm" Resource to anonymous requesters, to preserve privacy.

3116 **13.15 SVRs, Discoverability and OCF Endpoints**

3117 All implemented SVRs shall be "discoverable" (reference ISO/IEC 30118-1:2018, Policy Parameter  
 3118 clause 7.8.2.1.2).



3119 All implemented discoverable SVRs shall expose a Secure OCF Endpoint (e.g. CoAPS) (reference  
3120 ISO/IEC 30118-1:2018, clause 10).

3121 The "/oic/sec/doxm" Resource shall expose an Unsecure OCF Endpoint (e.g. CoAP) in RFOTM  
3122 (reference ISO/IEC 30118-1:2018, clause 10).

3123 **13.16 Additional Privacy Consideration for Core Resources**

3124 Unique immutable identifiers are a privacy consideration due to their potential for being used as a  
3125 tracking mechanism. These include the following Resources and Properties:

- 3126 – "/oic/d" Resource containing the "piid" Property.
- 3127 – "/oic/p" Resource containing the "pi" Property.

3128 These identifiers are unique values that are visible at various times throughout the Device lifecycle  
3129 by anonymous requestors. This implies any Client Device, including those with malicious intent,  
3130 are able to reliably obtain identifiers useful for building a log of activity correlated with a specific  
3131 Platform and Device.

3132 The "di" Property in the "/oic/d" Resource shall mirror that of the "deviceuuid" Property of the  
3133 "/oic/sec/doxm" Resource. The DOTS should provision an ACL policy that restricts access to the  
3134 "/oic/d" resource such that only authenticated Clients are able to obtain the "di" Property of "/oic/d"  
3135 Resource. See clause 13.1 for deviceuuid Property lifecycle requirements.

3136 Servers should expose a temporary, non-repeated, "piid" Property of "/oic/d" Resource Value upon  
3137 entering RESET Device state. Servers shall expose a persistent value via the "piid" Property of  
3138 "/oic/d" Property when the DOTS sets "devowneruuid" Property to a non-nil-UUID value. The DOTS  
3139 should provision an ACL policy on the "/oic/d" Resource such that only authenticated Clients are  
3140 able to obtain the "piid" Property of "/oic/d" Resource

3141 Servers should expose a temporary, non-repeated, "pi" Property value upon entering RESET  
3142 Device state. Servers shall expose a persistent value via the "pi" Property of the "/oic/p" Resource  
3143 when the DOTS sets "devowneruuid" Property to a non-nil-UUID value. The DOTS should provision  
3144 an ACL policy on the "/oic/p" Resource such that only authenticated Clients are able to obtain the  
3145 "pi" Property.

3146 Table 48 depicts Core Resource Properties Access Modes given various Device States.

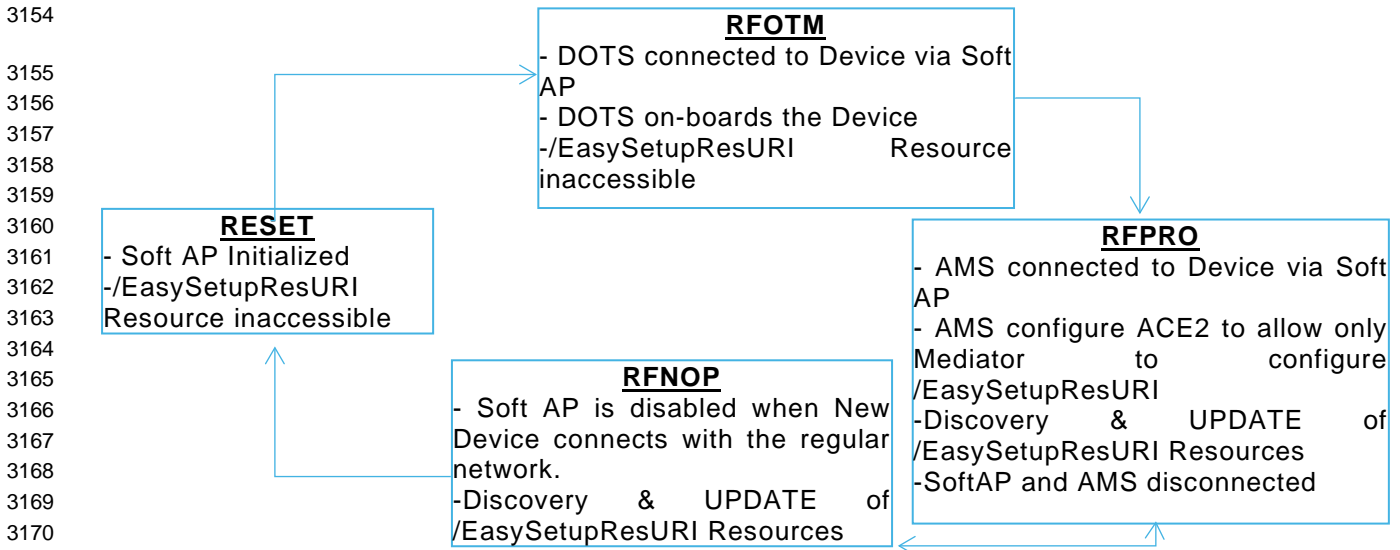
3147 **Table 48 – Core Resource Properties Access Modes given various Device States**

Resource Type	Property title	Property name	Value type	Access Mode		Behaviour
oic.wk.p	Platform ID	pi	oic.types-schema.uuid	All States	R	Server exposes a temporary random UUID when in RESET state.
oic.wk.d	Permanent Immutable ID	piid	oic.types-schema.uuid	All States	R	Server exposes a temporary random UUID when in RESET state.
oic.wk.d	Device Identifier	di	oic.types-schema.uuid	All states	R	/d di mirrors the value contained in "/doxm" "deviceuuid" in all device states.

3148 **13.17 Easy Setup Resource Device State**

3149 This clause only applies to a new Device that uses Easy Setup for ownership transfer as defined  
 3150 in OCF Wi-Fi Easy Setup. Easy Setup has no impact to new Devices that have a different way of  
 3151 connecting to the network i.e. DOTS and AMS don't use a Soft AP to connect to non-Easy Setup  
 3152 Devices.

3153 Figure 24 shows an example of Soft AP and Easy Setup Resource in different Device states.



3171 **Figure 24 – Example of Soft AP and Easy Setup Resource in different Device states**

3172 Device enters RFOTM Device state, Soft AP may be accessible in RFOTM and RFPRO Device's  
 3173 state.

3174 While it is reasonable for a user to expect that power cycling a new Device will turn on the Soft AP  
 3175 for Easy Setup during the initial setup, since that is potentially how it behaved on first boot, it is a  
 3176 security risk to make this the default behaviour of a device that remains unenrolled beyond a  
 3177 reasonable period after first boot.

3178 Therefore, the Soft AP for Easy Setup has several requirements to improve security:

- 3179 – Time availability of Easy Setup Soft AP should be minimised, and shall not exceed 30 minutes  
 3180 after Device factory reset RESET or first power boot, or when user initiates the Soft AP for Easy  
 3181 Setup.
- 3182 – If a new Device tried and failed to complete Easy Setup Enrolment immediately following the  
 3183 first boot, or after a factory reset, it may turn the Easy Setup Soft AP back on automatically for  
 3184 another 30 minutes upon being power cycled, provided that the power cycle occurs within 3  
 3185 hours of first boot or the most recent factory reset. If the user has initiated the Easy Setup Soft  
 3186 AP directly without a factory reset, it is not necessary to turn it back on if it was on immediately  
 3187 prior to power cycle, because the user obviously knows how to initiate the process manually.
- 3188 – After 3 hours from first boot or factory reset without successfully enrolling the device, the Soft  
 3189 AP should not turn back on for Easy Setup until another factory reset occurs, or the user initiates  
 3190 the Easy Setup Soft AP directly.
- 3191 – Easy Setup Soft AP may stay enabled during RFNOP, until the Mediator instructs the new  
 3192 Device to connect to the Enroller.
- 3193 – The Easy Setup Soft AP shall be disabled when the new Device successfully connects to the  
 3194 Enroller.

3195 – Once a new Device has successfully connected to the Enroller, it shall not turn the Easy Setup  
3196 Soft AP back on for Easy Setup Enrolment again unless the Device is factory reset, or the user  
3197 initiates the Easy Setup Soft AP directly.

3198 – Just Works OTM shall not be enabled on Devices which support Easy Setup.

3199 – The Soft AP shall be secured (e.g. shall not expose an open AP).

3200 – The Soft AP shall support a passphrase for connection by the Mediator, and the passphrase  
3201 shall be between and 8 and 64 ASCII printable characters. The passphrase may be printed on  
3202 a label, sticker, packaging etc., and may be entered by the user into the Mediator device.

3203 – The Soft AP should not use a common passphrase across multiple Devices. Instead, the  
3204 passphrase may be sufficiently unique per device, to prevent guessing of the passphrase by an  
3205 attacker with knowledge of the Device type, model, manufacturer, or any other information  
3206 discoverable through Device's exposed interfaces.

3207 The Enrollee shall support WPA2 security (i.e. shall list WPA2 in the "swat" Property of the  
3208 "/example/WiFiConfResURI" Resource), for potential selection by the Mediator in connecting the  
3209 Enrollee to the Enroller. The Mediator should select the best security available on the Enroller, for  
3210 use in connecting the Enrollee to the Enroller.

3211 The Enrollee may not expose any interfaces (e.g. web server, debug port, NCRs, etc.) over the  
3212 Soft AP, other than SVRs, and Resources required for Wi-Fi Easy Setup.

3213 The "/example/EasySetupResURI" Resource should not be discoverable in RFOTM or SRESET  
3214 state. After ownership transfer process is completed with the DOTS, and the Device enters in  
3215 RFPRO Device state, the "/example/EasySetupResURI" may be Discoverable.

3216 The OTM CoAPS session may be used by Mediator for connection over Soft AP for ownership  
3217 transfer and initial Easy Setup provisioning. SoftAP or regular network connection may be used by  
3218 AMS for "/oic/sec/acl2" Resource provisioning in RFPRO state. The CoAPS session authentication  
3219 and encryption is already defined in the Security spec.

3220 In RFPRO state, AMS is expected to configure ACL2 Resource on the Device with ACE2 for  
3221 following Resources to be only configurable by the Mediator with permission to UPDATE or  
3222 RETRIEVE access:

3223 – "/example/EasySetupResURI"  
3224 – "/example/WifiConfResURI"  
3225 – "/example/DevConfResURI"

3226 An ACE2 granting RETRIEVE or UPDATE access to the Easy Setup Resource

```

3227 {
3228     "subject": { "uuid": "<insert-UUID-of-Mediator>" },
3229     "resources": [
3230         { "href": "/example/EasySetupResURI" },
3231         { "href": "/example/WiFiConfResURI" },
3232         { "href": "/example/DevConfResURI" },
3233     ],
3234     "permission": 6 // RETRIEVE (2) or UPDATE and RETRIEVE(6)
3235 }
```

3236 ACE2 may be re-configured after Easy Setup process. These ACE2s should be installed prior to  
3237 the Mediator performing any RETRIEVE/UPDATE operations on these Resources.

3238 In RFPRO or RFNOP, the Mediator should discover /EasySetupResURI Resources and UPDATE  
3239 these Resources. The Mediator may UPDATE /EasySetupResURI resources in RFNOP Device  
3240 state.

3241 A Mediator shall be hosted on an OCF Device.

3242 **14 Security Hardening Guidelines/ Execution Environment Security**

3243 **14.1 Preamble**

3244 This is an informative clause. Many TGs in OCF have security considerations for their protocols  
3245 and environments. These security considerations are addressed through security mechanisms  
3246 specified in the security documents for OCF. However, effectiveness of these mechanisms depends  
3247 on security robustness of the underlying hardware and software Platform. This clause defines the  
3248 components required for execution environment security.

3249 **14.2 Execution Environment Elements**

3250 **14.2.1 Execution Environment Elements General**

3251 Execution environment within a computing Device has many components. To perform security  
3252 functions in a robustness manner, each of these components has to be secured as a separate  
3253 dimension. For instance, an execution environment performing AES cannot be considered secure  
3254 if the input path entering keys into the execution engine is not secured, even though the partitions  
3255 of the CPU, performing the AES encryption, operate in isolation from other processes. Different  
3256 dimensions referred to as elements of the execution environment are listed below. To qualify as a  
3257 secure execution environment (SEE), the corresponding SEE element must qualify as secure.

- 3258 – (Secure) Storage
- 3259 – (Secure) Execution engine
- 3260 – (Trusted) Input/output paths
- 3261 – (Secure) Time Source/clock
- 3262 – (Random) number generator
- 3263 – (Approved) cryptographic algorithms
- 3264 – Hardware Tamper (protection)

3265 NOTE Software security practices (such as those covered by OWASP) are outside scope of this document, as  
3266 development of secure code is a practice to be followed by the open source development community. This document will  
3267 however address the underlying Platform assistance required for executing software. Examples are secure boot and  
3268 secure software upgrade.

3269 Each of the elements above are described in the clauses 14.2.2, 14.2.3, 14.2.4, 14.2.5, 14.2.6,  
3270 14.2.7.

3271 **14.2.2 Secure Storage**

3272 **14.2.2.1 Secure Storage General**

3273 Secure storage refers to the physical method of housing sensitive or confidential data ("Sensitive  
3274 Data"). Such data could include but not be limited to symmetric or asymmetric private keys,  
3275 certificate data, OCF Security Domain access credentials, or personal user information. Sensitive  
3276 Data requires that its integrity be maintained, whereas Critical Sensitive Data requires that both its  
3277 integrity and confidentiality be maintained.

3278 It is strongly recommended that IoT Device makers provide reasonable protection for Sensitive  
3279 Data so that it cannot be accessed by unauthorized Devices, groups or individuals for either  
3280 malicious or benign purposes. In addition, since Sensitive Data is often used for authentication and  
3281 encryption, it must maintain its integrity against intentional or accidental alteration.

3282 A partial list of Sensitive Data is outlined in Table 49:

**Table 49 – Examples of Sensitive Data**

<b>Data</b>	<b>Integrity protection</b>	<b>Confidentiality protection</b>
Owner PSK (Symmetric Keys)	Yes	Yes
Service provisioning keys	Yes	Yes
Asymmetric Private Keys	Yes	Yes
Certificate Data and Signed Hashes	Yes	Not required
Public Keys	Yes	Not required
Access credentials (e.g. SSID, passwords, etc.)	Yes	Yes
ECDH/ECDH Dynamic Shared Key	Yes	Yes
Root CA Public Keys	Yes	Not required
Device and Platform IDs	Yes	Not required
Easy Setup Resources	Yes	Yes
Access Token	Yes	Yes

3284 Exact method of protection for secure storage is implementation specific, but typically combinations  
3285 of hardware and software methods are used.

#### 3286 **14.2.2.2 Hardware Secure Storage**

3287 Hardware secure storage is recommended for use with critical Sensitive Data such as symmetric  
3288 and asymmetric private keys, access credentials, and personal private data. Hardware secure  
3289 storage most often involves semiconductor-based non-volatile memory ("NVRAM") and includes  
3290 countermeasures for protecting against unauthorized access to Critical Sensitive Data.

3291 Hardware-based secure storage not only stores Sensitive Data in NVRAM, but also provides  
3292 protection mechanisms to prevent the retrieval of Sensitive Data through physical and/or electronic  
3293 attacks. It is not necessary to prevent the attacks themselves, but an attempted attack should not  
3294 result in an unauthorized entity successfully retrieving Sensitive Data.

3295 Protection mechanisms should provide JIL Moderate protection against access to Sensitive Data  
3296 from attacks that include but are not limited to:

- 3297 1) Physical decapping of chip packages to optically read NVRAM contents
- 3298 2) Physical probing of decapped chip packages to electronically read NVRAM contents
- 3299 3) Probing of power lines or RF emissions to monitor voltage fluctuations to discern the bit patterns  
3300 of Critical Sensitive Data
- 3301 4) Use of malicious software or firmware to read memory contents at rest or in transit within a  
3302 microcontroller
- 3303 5) Injection of faults that induce improper Device operation or loss or alteration of Sensitive Data

#### 3304 **14.2.2.3 Software Storage**

3305 It is generally NOT recommended to rely solely on software and unsecured memory to store  
3306 Sensitive Data even if it is encrypted. Critical Sensitive Data such as authentication and encryption  
3307 keys should be housed in hardware secure storage whenever possible.

3308 Sensitive Data stored in volatile and non-volatile memory shall be encrypted using acceptable  
3309 algorithms to prevent access by unauthorized parties through methods described in 14.2.2.2.

#### 3310 **14.2.2.4 Additional Security Guidelines and Best Practices**

3311 Some general practices that can help ensure that Sensitive Data is not compromised by various  
3312 forms of security attacks:

- 3313 1) FIPS Random Number Generator ("RNG") – Insufficient randomness or entropy in the RNG  
3314 used for authentication challenges can substantially degrade security strength. For this reason,  
3315 it is recommended that a FIPS 800-90A-compliant RNG with a certified noise source be used  
3316 for all authentication challenges.
- 3317 2) Secure download and boot – To prevent the loading and execution of malicious software, where  
3318 it is practical, it is recommended that Secure Download and Secure Boot methods that  
3319 authenticate a binary's source as well as its contents be used.
- 3320 3) Deprecated algorithms – Algorithms included but not limited to the list below are considered  
3321 unsecure and shall not be used for any security-related function:
  - 3322 a) SHA-1
  - 3323 b) MD5
  - 3324 c) RC4
  - 3325 d) RSA 1024
- 3326 4) Encrypted transmission between blocks or components – Even if critical Sensitive Data is  
3327 stored in Secure Storage, any use of that data that requires its transmission out of that Secure  
3328 Storage should be encrypted to prevent eavesdropping by malicious software within an  
3329 MCU/MPU.
- 3330 5) It is recommended to avoid using wildcard in Subject Id ("\*"), when setting up "/oic/sec/cred"  
3331 Resource entries, since this opens up an identity spoofing opportunity.
- 3332 6) Device vendor understands that it is the Device vendor's responsibility to ensure the Device  
3333 meets security requirements for its intended uses. As an example, IoTivity is a reference  
3334 implementation intended to be used as a basis for a product, but IoTivity has not undergone  
3335 3rd party security review, penetration testing, etc. Any Device based on IoTivity should undergo  
3336 appropriate penetration testing and security review prior to sale or deployment.
- 3337 7) Device vendor agrees to publish the expected support lifetime for the Device to OCF and to  
3338 consumers. Changes should be made to a public and accessible website. Expectations should  
3339 be clear as to what will be supported and for how long the Device vendor expects to support  
3340 security updates to the software, operating system, drivers, networking, firmware and hardware  
3341 of the device.
- 3342 8) Device vendor has not implemented test or debug interfaces on the Device which are operable  
3343 or which can be enabled which might present an attack vector on the Device which circumvents  
3344 the interface-level security or access policies of the Device.
- 3345 9) Device vendor understands that if an application running on the Device has access to  
3346 cryptographic elements such as the private keys or Ownership Credential, then those elements  
3347 have become vulnerable. If the Device vendor is implementing a Bridge, an OBT, or a Device  
3348 with access to the Internet beyond the local network, the execution of critical functions should  
3349 take place within a Trusted or Secure Execution Environment (TEE/SEE).
- 3350 10) Any PINs or fixed passphrases used for onboarding, Wi-Fi Easy Setup, SoftAP management or  
3351 access, or other security-critical function, should be sufficiently unique (do not duplicate  
3352 passphrases. The creation of these passphrases or PINS should not be algorithmically  
3353 deterministic nor should they use insufficient entropy in their creation.
- 3354 11) Ensure that there are no remaining "VENDOR\_TODO" items in the source code.

3355 12) If the implementation of this document uses the "Just Works" onboarding method, understand  
3356 that there is a man-in-the-middle vulnerability during the onboarding process where a malicious  
3357 party could intercept messages between the device being onboarded and the OBT and could  
3358 persist, acting as an intermediary with access to message traffic, during the lifetime of that  
3359 onboarded device. The recommended best practice would be to use an alternate ownership  
3360 transfer method (OTM) instead of "Just Works".

3361 13) It is recommended that at least one static and dynamic analysis tool<sup>1</sup> be applied to any  
3362 proposed major production release of the software before its release, and any vulnerabilities  
3363 resolved.

### 3364 **14.2.3 Secure execution engine**

3365 Execution engine is the part of computing Platform that processes security functions, such as  
3366 cryptographic algorithms or security protocols (e.g. DTLS). Securing the execution engine requires  
3367 the following

3368 – Isolation of execution of sensitive processes from unauthorized parties/ processes. This  
3369 includes isolation of CPU caches, and all of execution elements that needed to be considered  
3370 as part of trusted (crypto) boundary.

3371 – Isolation of data paths into and out of execution engine. For instance, both unencrypted but  
3372 sensitive data prior to encryption or after decryption, or cryptographic keys used for  
3373 cryptographic algorithms, such as decryption or signing. See clause 14.2.4 for more details.

### 3374 **14.2.4 Trusted input/output paths**

3375 Paths/ ports used for data entry into or export out of trusted/ crypto-boundary needs to be protected.  
3376 This includes paths into and out secure execution engine and secure memory.

3377 Path protection can be both hardware based (e.g. use of a privileged bus) or software based (using  
3378 encryption over an untrusted bus).

### 3379 **14.2.5 Secure clock**

3380 Many security functions depend on time-sensitive credentials. Examples are time stamped  
3381 Kerberos tickets, OAuth tokens, X.509 certificates, OSCP response, software upgrades, etc. Lack  
3382 of secure source of clock can mean an attacker can modify the system clock and fool the validation  
3383 mechanism. Thus an SEE needs to provide a secure source of time that is protected from tampering.  
3384 Trustworthiness from security robustness standpoint is not the same as accuracy. Protocols such  
3385 as NTP can provide rather accurate time sources from the network, but are not immune to attacks.  
3386 A secure time source on the other hand can be off by seconds or minutes depending on the time-  
3387 sensitivity of the corresponding security mechanism. Secure time source can be external as long  
3388 as it is signed by a trusted source and the signature validation in the local Device is a trusted  
3389 process (e.g. backed by secure boot).

### 3390 **14.2.6 Approved algorithms**

3391 An important aspect of security of the entire ecosystem is the robustness of publicly vetted and  
3392 peer-reviewed (e.g. NIST-approved) cryptographic algorithms. Security is not achieved by  
3393 obscurity of the cryptographic algorithm. To ensure both interoperability and security, not only  
3394 widely accepted cryptographic algorithms must be used, but also a list of approved cryptographic  
3395 functions must be specified explicitly. As new algorithms are NIST approved or old algorithms are  
3396 deprecated, the list of approved algorithms must be maintained by OCF. All other algorithms (even  
3397 if they deemed stronger by some parties) must be considered non-approved.

3398 The set of algorithms to be considered for approval are algorithms for

3399 – Hash functions

---

<sup>1</sup> A general discussion of analysis tools can be found here: <https://www.ibm.com/developerworks/library/se-static/>



- 3400 – Signature algorithms
- 3401 – Encryption algorithms
- 3402 – Key exchange algorithms
- 3403 – Pseudo Random functions (PRF) used for key derivation
- 3404 This list will be included in this or a separate security robustness rules document and must be
- 3405 followed for all security specifications within OCF.

#### 3406 **14.2.7 Hardware tamper protection**

3407 Various levels of hardware tamper protection exist. We borrow FIPS 140-2 terminology (not  
3408 requirements) regarding tamper protection for cryptographic module

- 3409 – Production-grade (lowest level): this means components that include conformal sealing coating
- 3410 applied over the module’s circuitry to protect against environmental or other physical damage.
- 3411 This does not however require zeroization of secret material during physical maintenance. This
- 3412 definition is borrowed from FIPS 140-2 security level 1.
- 3413 – Tamper evident/proof (mid-level), This means the Device shows evidence (through covers,
- 3414 enclosures, or seals) of an attempted physical tampering. This definition is borrowed from FIPS
- 3415 140-2 security level 2.
- 3416 – Tamper resistance (highest level), this means there is a response to physical tempering that
- 3417 typically includes zeroization of sensitive material on the module. This definition is borrowed
- 3418 from FIPS 140-2 security level 3.

3419 It is difficult of specify quantitative certification test cases for accreditation of these levels. Content  
3420 protection regimes usually talk about different tools (widely available, specialized and professional  
3421 tools) used to circumvent the hardware protections put in place by manufacturing. If needed, OCF  
3422 can follow that model, if and when OCF engage in distributing sensitive key material (e.g. PKI) to  
3423 its members.

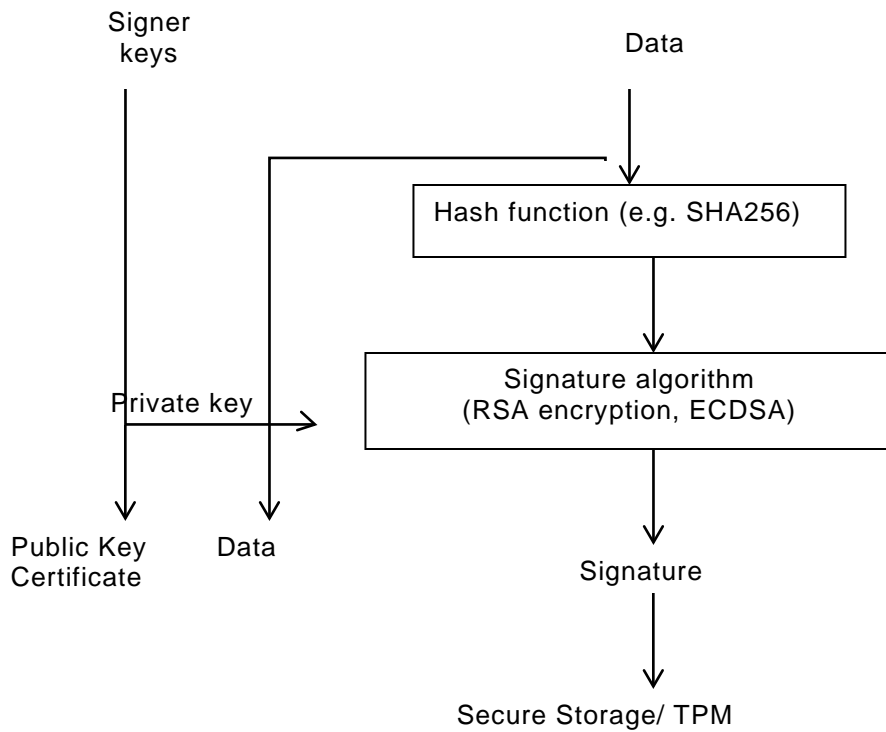
### 3424 **14.3 Secure Boot**

#### 3425 **14.3.1 Concept of software module authentication**

3426 In order to ensure that all components of a Device are operating properly and have not been  
3427 tampered with, it is best to ensure that the Device is booted properly. There may be multiple stages  
3428 of boot. The end result is an application running on top an operating system that takes advantage  
3429 of memory, CPU and peripherals through drivers.

3430 The general concept is that each software module is invoked only after cryptographic integrity  
3431 verification is complete. The integrity verification relies on the software module having been hashed  
3432 (e.g. SHA\_1, SHA\_256) and then signed with a cryptographic signature algorithm with (e.g. RSA),  
3433 with a key that only a signing authority has access to.

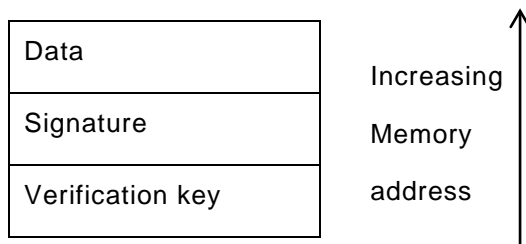
3434 Figure 25 depicts software module authentication.



3435 **Figure 25 – Software Module Authentication**

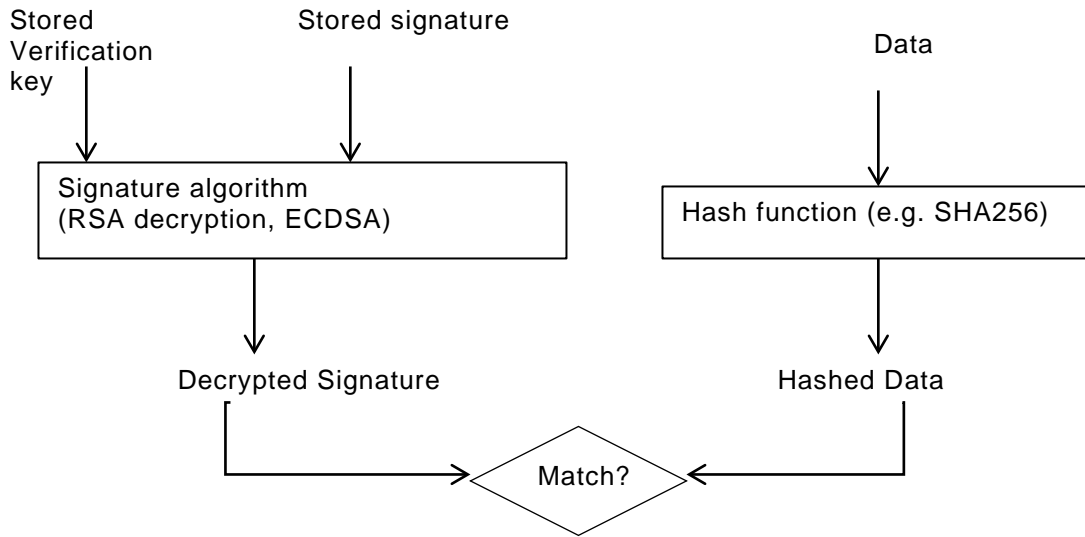
3436 After the data is signed with the signer’s signing key (a private key), the verification key (the public  
 3437 key corresponding to the private signing key) is provided for later verification. For lower level  
 3438 software modules, such as bootloaders, the signatures and verification keys are inserted inside  
 3439 tamper proof memory, such as one-time programmable memory or TPM. For higher level software  
 3440 modules, such as application software, the signing is typically performed according to the PKCS#7  
 3441 format IETF RFC 2315, where the signedData format includes both indications for signature  
 3442 algorithm, hash algorithm as well as the signature verification key (or certificate). Secure boot does  
 3443 not require use of PKCS#7 format.

3444 Figure 26 depicts verification software module.



3445 **Figure 26 – Verification Software Module**

3446 As shown in Figure 27. the verification module first decrypts the signature with the verification key  
 3447 (public key of the signer). The verification module also calculates a hash of the data and then  
 3448 compares the decrypted signature (the original) with the hash of data (actual) and if the two values  
 3449 match, the software module is authentic.



3450 **Figure 27 – Software Module Authenticity**

3450

3451 **14.3.2 Secure Boot process**

3452 Depending on the Device implementation, there may be several boot stages. Typically, in a PC/  
 3453 Linux type environment, the first step is to find and run the BIOS code (first-stage bootloader) to  
 3454 find out where the boot code is and then run the boot code (second-stage boot loader). The second  
 3455 stage bootloader is typically the process that loads the operating system (Kernel) and transfers the  
 3456 execution to the where the Kernel code is. Once the Kernel starts, it may load external Kernel  
 3457 modules and drivers.

3458 When performing a secure boot, it is required that the integrity of each boot loader is verified before  
 3459 executing the boot loader stage. As mentioned, while the signature and verification key for the  
 3460 lowest level bootloader is typically stored in tamper-proof memory, the signature and verification  
 3461 key for higher levels should be embedded (but attached in an easily accessible manner) in the data  
 3462 structures software.

3463 **14.3.3 Robustness Requirements**

3464 **14.3.3.1 Robustness General**

3465 To qualify as high robustness secure boot process, the signature and hash algorithms shall be one  
 3466 of the approved algorithms, the signature values and the keys used for verification shall be stored  
 3467 in secure storage and the algorithms shall run inside a secure execution environment and the keys  
 3468 shall be provided the SEE over trusted path.

3469 **14.3.3.2 Next steps**

3470 Develop a list of approved algorithms and data formats

3471 **14.4 Attestation**

3472 **14.5 Software Update**

3473 **14.5.1 Overview:**

3474 The Device lifecycle does not end at the point when a Device is shipped from the manufacturer;  
 3475 the distribution, retailing, purchase, installation/onboarding, regular operation, maintenance and  
 3476 end-of-life stages for the Device remain outstanding. It is possible for the Device to require update

3477 during any of these stages, although the most likely times are during onboarding, regular operation  
 3478 and maintenance. The aspects of the software include, but are not limited to, firmware, operating  
 3479 system, networking stack, application code, drivers, etc.

3480 **14.5.2 Recognition of Current Differences**

3481 Different manufacturers approach software update utilizing a collection of tools and strategies:  
 3482 over-the-air or wired USB connections, full or partial replacement of existing software, signed and  
 3483 verified code, attestation of the delivery package, verification of the source of the code, package  
 3484 structures for the software, etc.

3485 It is recommended that manufacturers review their processes and technologies for compliance with  
 3486 industry best-practices that a thorough security review of these takes place and that periodic review  
 3487 continue after the initial architecture has been established.

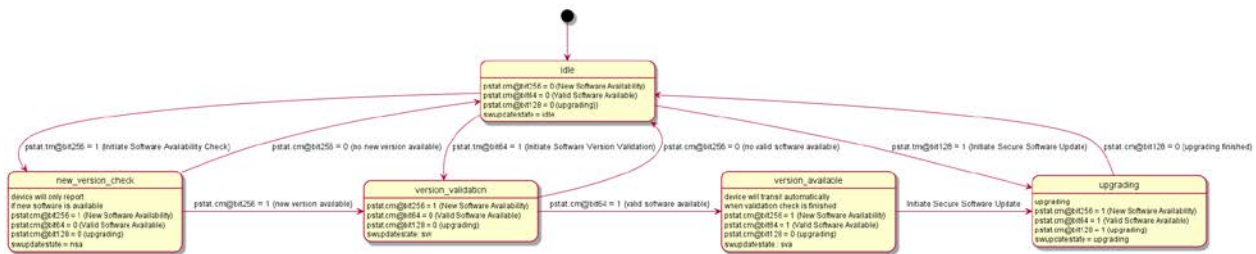
3488 This document applies to software updates as recommended to be implemented by OCF Devices;  
 3489 it does not have any bearing on the above-mentioned alternative proprietary software update  
 3490 mechanisms. The described steps are being triggered by an OCF Client, the actual implementation  
 3491 of the steps and how the software package is downloaded and upgraded is vendor specific.

3492 The triggers that can be invoked from OCF clients can perform:

- 3493 1) Check if new software is available
- 3494 2) Download and verify the integrity of the software package
- 3495 3) Install the verified software package

3496 The triggers are not sequenced, each trigger can be invoked individually.

3497 The state of the transitions of software update is in Figure 28.



3498 **Figure 28 – State transitioning diagram for software download**

3499 **Table 50 – Description of the software update bits**

Bit	TM property	CM property
Bit 9	Initiate Software Availability Check	New Software Available
Bit 7	Initiate Software Version Validation	Valid Software Available
Bit 8	Initiate Secure Software Update	Upgrading

3500  
 3501  
 3502  
 3503 **14.5.2.1 Checking availability of new software**

3504 Setting the Initiate Software Availability Check bit in the "/oic/sec/pstat.tm" Property (see Table 37  
 3505 of clause 13.8) indicates a request to initiate the process to check if new software is available, e.g.  
 3506 the process whereby the Device checks if a newer software version is available on the external

3507 endpoint. Once the Device has determined if a newer software version is available, it sets the  
3508 Initiate Software Availability Check bit in the "/oic/sec/pstat.cm" Property to 1 (TRUE), indicating  
3509 that new software is available or to 0 (FALSE) if no newer software version is available, See also  
3510 Table 50 where the bits in property TM indicates that the action is initiated and the CM bits are  
3511 indicating the result of the action. The Device receiving this trigger is not downloading and not  
3512 validating the software to determine if new software is available. The version check is determined  
3513 by the current software version and the software version on the external endpoint. The  
3514 determination if a software package is newer is vendor defined.

### 3515 **14.5.3 Software Version Validation**

3516 Setting the Initiate Software Version Validation bit in the "/oic/sec/pstat.tm" Property (see Table 37  
3517 of 13.8) indicates a request to initiate the software version validation process, the process whereby  
3518 the Device validates the software (including firmware, operating system, Device drivers, networking  
3519 stack, etc.) against a trusted source to see if, at the conclusion of the check, the software update  
3520 process will need to be triggered (see clause 14.5.4). When the Initiate Software Version Validation  
3521 bit of "/oic/sec/pstat.tm" is set to 1 (TRUE) by a sufficiently privileged Client, the Device sets the  
3522 "/oic/sec/pstat.cm" Initiate Software Version Validation bit to 0 and initiates a software version  
3523 check. Once the Device has determined if a valid software is available, it sets the Initiate Software  
3524 Version Validation bit in the "/oic/sec/pstat.cm" Property to 1 (TRUE) if an update is available or 0  
3525 (FALSE) if no update is available. To signal completion of the Software Version Validation process,  
3526 the Device sets the Initiate Software Version Validation bit in the "/oic/sec/pstat.tm" Property back  
3527 to 0 (FALSE). If the Initiate Software Version Validation bit of "/oic/sec/pstat.tm" is set to 0 (FALSE)  
3528 by a Client, it has no effect on the validation process. The Software Version Validation process can  
3529 download the software from the external endpoint to verify the integrity of the software package.

### 3530 **14.5.4 Software Update**

3531 Setting the Initiate Secure Software Update bit in the "/oic/sec/pstat.tm" Property (see Table 37 of  
3532 clause 13.8) indicates a request to initiate the software update process. When the Initiate Secure  
3533 Software Update bit of "/oic/sec/pstat.tm" is set to 1 (TRUE) by a sufficiently privileged Client, the  
3534 Device sets the "/oic/sec/pstat.cm" Initiate Software Version Validation bit to 0 and initiates a  
3535 software update process. Once the Device has completed the software update process, it sets the  
3536 Initiate Secure Software Update bit in the "/oic/sec/pstat.cm" Property to 1 (TRUE) if/when the  
3537 software was successfully updated or 0 (FALSE) if no update was performed. To signal completion  
3538 of the Secure Software Update process, the Device sets the Initiate Secure Software Update bit in  
3539 the "/oic/sec/pstat.tm" Property back to 0 (FALSE). If the Initiate Secure Software Update bit of  
3540 "/oic/sec/pstat.tm" is set to 0 (FALSE) by a Client, it has no effect on the update process.

#### 3541 **14.5.4.1 State of Device after software update**

3542 The state of all resources implemented in the Device should be the same as after boot, meaning  
3543 that the software update is not resetting user data and retaining a correct state.

3544 User data of a Device is defined as:

- 3545 – Retain the SVR states, e.g. the on boarded state, registered clients.
- 3546 – Retain all created resources
- 3547 – Retain all stored data of a resource
- 3548 – For example the preferences stored for the brewing resource ("oic.r.brewing").

### 3549 **14.5.5 Recommended Usage**

3550 The Initiate Secure Software Update bit of "/oic/sec/pstat.tm" should only be set by a Client after  
3551 the Initiate Software Version Validation check is complete.

3552 The process of updating Device software may involve state changes that affect the Device  
3553 Operational State ("/oic/sec/pstat.dos"). Devices with an interest in the Device(s) being updated

3554 should monitor "/oic/sec/pstat.dos" and be prepared for pending software update(s) to affect Device  
3555 state(s) prior to completion of the update.

3556 The Device itself may indicate that it is autonomously initiating a software version check/update or  
3557 that a check/update is complete by setting the "pstat.tm" and "pstat.cm" Initiate Software Version  
3558 Validation and Secure Software Update bits when starting or completing the version check or  
3559 update process. As is the case with a Client-initiated update, Clients can be notified that an  
3560 autonomous version check or software update is pending and/or complete by observing pstat  
3561 resource changes.

3562 The "oic.r.softwareupdate" Resource Type specifies additional features to control the software  
3563 update process see core specification.

#### 3564 **14.6 Non-OCF Endpoint interoperability**

#### 3565 **14.7 Security Levels**

3566 Security Levels are a way to differentiate Devices based on their security criteria. This need for  
3567 differentiation is based on the requirements from different verticals such as industrial and health  
3568 care and may extend into smart home. This differentiation is distinct from Device classification  
3569 (e.g. IETF RFC 7228)

3570 These categories of security differentiation may include, but is not limited to:

- 3571 1) Security Hardening
- 3572 2) Identity Attestation
- 3573 3) Certificate/Trust
- 3574 4) Onboarding Technique
- 3575 5) Regulatory Compliance
  - 3576 a) Data at rest
  - 3577 b) Data in transit
- 3578 6) Cipher Suites – Crypto Algorithms & Curves
- 3579 7) Key Length
- 3580 8) Secure Boot/Update

3581 In the future security levels can be used to define interoperability.

3582 The following applies to the OCF Security Specification 1.1:

3583 The current document does not define any other level beyond Security Level 0. All Devices will be  
3584 designated as Level 0. Future versions may define additional levels.

3585 Additional comments:

- 3586 – The definition of a given security level will remain unchanged between versions of the document.
- 3587 – Devices that meet a given level may, or may not, be capable of upgrading to a higher level.
- 3588 – Devices may be evaluated and re-classified at a higher level if it meets the requirements of the  
3589 higher level (e.g. if a Device is manufactured under the 1.1 version of the document, and a later  
3590 document version defines a security level 1, the Device could be evaluated and classified as  
3591 level 1 if it meets level 1 requirements).
- 3592 – The security levels may need to be visible to the end user.

3593 **14.8 Security Profiles**

3594 **14.8.1 Security Profiles General**

3595 Security Profiles are a way to differentiate OCF Devices based on their security criteria. This need  
3596 for differentiation is based on the requirements from different verticals such as industrial and health  
3597 care and may extend into smart home. This differentiation is distinct from device classification (e.g.  
3598 IETF RFC 7228)

3599 These categories of security differentiation may include, but is not limited to:

- 3600 1) Security Hardening and assurances criteria
- 3601 2) Identity Attestation
- 3602 3) Certificate/Trust
- 3603 4) Onboarding Technique
- 3604 5) Regulatory Compliance
  - 3605 a) Data at rest
  - 3606 b) Data in transit
- 3607 6) Cipher Suites – Crypto Algorithms & Curves
- 3608 7) Key Length
- 3609 8) Secure Boot/Update

3610 Each Security Profile definition must specify the version or versions of the OCF Security  
3611 Specification(s) that form a baseline set of normative requirements. The profile definition may  
3612 include security requirements that supersede baseline requirements (not to relax security  
3613 requirements).

3614 Security Profiles have the following properties:

- 3615 – A given profile definition is not specific to the version of the document that defines it. For  
3616 example, the profile may remain constant for subsequent OCF Security Specification versions.
- 3617 – A specific OCF Device and platform combination may be used to satisfy the security profile.
- 3618 – Profiles may have overlapping criteria; hence it may be possible to satisfy multiple profiles  
3619 simultaneously.
- 3620 – An OCF Device that satisfied a profile initially may be re-evaluated at a later time and found to  
3621 satisfy a different profile (e.g. if a device is manufactured under the 1.1 version of the document,  
3622 and a later document version defines a security profile Black, the device could be evaluated  
3623 and classified as profile Black if it meets profile Black requirements).
- 3624 – A machine-readable representation of compliance results specifically describing profiles  
3625 satisfied may be used to facilitate OCF Device onboarding. (e.g. a manufacturer certificate or  
3626 manifest may contain security profiles attributes).

3627 **14.8.2 Identification of Security Profiles (Normative)**

3628 **14.8.2.1 Security Profiles in Prior Documents**

3629 OCF Devices conforming to versions of the OCF Security Specifications where Security Profiles  
3630 Resource was not defined may be presumed to satisfy the "sp-baseline-v0" profile (defined in  
3631 14.8.3.3) or may be regarded as unspecified. If Security Profile is unspecified, the Client may use  
3632 the OCF Security Specification version to characterize expected security behaviour.

3633 **14.8.2.2 Security Profile Resource Definition**

3634 The "/oic/sec/sp" Resource is used by the OCF Device to show which OCF Security Profiles the  
3635 OCF Device is capable of supporting and which are authorized for use by the OCF Security Domain

3636 owner. Properties of the Resource identify which OCF Security Profile is currently operational. The  
 3637 ocfSecurityProfileOID value type shall represent OID values and may reference an entry in the form  
 3638 of strings (UTF-8).

3639 "/oic/sec/sp" Resource is defined in Table 51.

3640 **Table 51 – Definition of the "/oic/sec/sp" Resource**

Fixed URI	Resource Type Title	Resource Type ID ("rt" value)	OCF Interfaces	Description	Related Functional Interaction
/oic/sec/sp	Security Profile Resource Definition	oic.r.sp	oic.if.baseline	Resource specifying supported and current security profile(s)	Discoverable

3641 Table 52 defines the Properties of "/oic/sec/sp" Resource.

3642 **Table 52 – Properties of the "/oic/sec/sp" Resource**

Property Title	Property Name	Value Type	Value Rule	Access Mode	Mandatory	Description
Supported Security Profiles	supportedprofiles	ocfSecurityProfileOID	array	RW	Yes	Array of supported Security Profiles (e.g. ["1.3.6.1.4.1.51414.0.0.2.0","1.3.6.1.4.1.51414.0.0.3.0"])
SecurityProfile	currentprofile	ocfSecurityProfileOID	N/A	RW	Yes	Currently active Security Profile (e.g. "1.3.6.1.4.1.51414.0.0.3.0")

3643 The following OIDs are defined to uniquely identify Security Profiles. Future Security Profiles or  
 3644 changes to existing Security Profiles may result in a new ocfSecurityProfileOID.

3645 id-OCF OBJECT IDENTIFIER ::= { iso(1) identified-organization(3) dod(6) internet(1)  
 3646 private(4) enterprise(1) OCF(51414) }

3647  
 3648 id-ocfSecurity OBJECT IDENTIFIER ::= { id-OCF 0 }

3649  
 3650 id-ocfSecurityProfile ::= { id-ocfSecurity 0 }

3651  
 3652 sp-undefined ::= OBJECT IDENTIFIER { id-ocfSecurityProfile 0 }

3653 --The Security Profile is not specified

3654 sp-baseline ::= OBJECT IDENTIFIER { id-ocfSecurityProfile 1 }

3655 --This specifies the OCF Baseline Security Profile(s)

3656 sp-black ::= OBJECT IDENTIFIER { id-ocfSecurityProfile 2 }

3657 --This specifies the OCF Black Security Profile(s)

3658 sp-blue ::= OBJECT IDENTIFIER { id-ocfSecurityProfile 3 }

3659 --This specified the OCF Blue Security Profile(s)

3660 sp-purple ::= OBJECT IDENTIFIER { id-ocfSecurityProfile 4 }

3661 --This specifies the OCF Purple Security Profile(s)

3662  
 3663 --versioned Security Profiles

3664 sp-undefined-v0 ::= ocfSecurityProfileOID (id-sp-undefined 0)

3665 --v0 of unspecified security profile, "1.3.6.1.4.1.51414.0.0.0.0"

3666 sp-baseline-v0 ::= ocfSecurityProfileOID {id-sp-baseline 0}

3667 --v0 of baseline security profile, "1.3.6.1.4.1.51414.0.0.1.0"

3668 sp-black-v0 ::= ocfSecurityProfileOID {id-sp-black 0}

3669 --v0 of black security profile, "1.3.6.1.4.1.51414.0.0.2.0"

3670 sp-blue-v0 ::= ocfSecurityProfileOID {id-sp-blue 0}

3671 --v0 of blue security profile, "1.3.6.1.4.1.51414.0.0.3.0"

3672 sp-purple-v0 ::= ocfSecurityProfileOID {id-sp-purple 0}

3673 --v0 of purple security profile, "1.3.6.1.4.1.51414.0.0.4.0"

3674

3675 ocfSecurityProfileOID ::= UTF8String



3676

3677 **14.8.3 Security Profiles**

3678 **14.8.3.1 Security Profiles General**

3679 The Security Profiles Resource shall be pre-populated with manufacturer default values (Refer to  
3680 the Security Profile clauses for additional details).

3681 The OCF Conformance criteria may require vendor attestation that establishes the expected  
3682 environment in which the OCF Device is hosted (Refer to the Security Profile clauses for specific  
3683 requirements).

3684 **14.8.3.2 Security Profile Unspecified (sp-unspecified-v0)**

3685 The Security Profile "sp-unspecified-v0" is reserved for future use.

3686 **14.8.3.3 Security Profile Baseline v0 (sp-baseline-v0)**

3687 The Security Profile "sp-baseline-v0" is defined for all OCF Security Specification versions where  
3688 the "/oic/sec/sp" Resource is defined. All Devices shall include the "sp-baseline-v0" OID in the  
3689 "supportedprofiles" Property of the "/oic/sec/sp" Resource.

3690 It indicates the OCF Device satisfies the normative security requirements for this document.

3691 When a device supports the baseline profile, the "supportedprofiles" Property shall contain sp-  
3692 baseline-v0, represented by the OID string "1.3.6.1.4.1.51414.0.0.1.0", and may contain other  
3693 profiles.

3694 When a manufacturer makes sp-baseline-v0 the default, by setting the "currentprofile" Property to  
3695 "1.3.6.1.4.1.51414.0.0.1.0", the "supportedprofiles" Property shall contain sp-baseline-v0.

3696 **14.8.3.4 Security Profile Black (sp-black-v0)**

3697 **14.8.3.4.1 Black Profile General**

3698 The need for Security Profile Black v0 is to support devices and manufacturers who wish to certify  
3699 their devices meeting this specific set of security criteria. A Device may satisfy the Black  
3700 requirements as well as requirements of other profiles, the Black Security Profile is not necessarily  
3701 mutually exclusive with other Security Profiles unless those requirements conflict with the explicit  
3702 requirements of the Black Security Profile.

3703 **14.8.3.4.2 Devices Targeted for Security Profile Black v0**

3704 Security Profile Black devices could include any device a manufacturer wishes to certify at this  
3705 profile, but healthcare devices and industrial devices with additional security requirements are the  
3706 initial target. Additionally, manufacturers of devices at the edge of the network (or fog), or devices  
3707 with exceptional profiles of trust bestowed upon them, may wish to certify at this profile; these types  
3708 of devices may include, but are not limited to the following:

3709 – Bridges (Mapping devices between ecosystems handling virtual devices from different  
3710 ecosystems)

3711 – Resource Directories (Devices trusted to manage OCF Security Domain resources)

3712 – Remote Access (Devices which have external access but can also act within the OCF Security  
3713 Domain)

3714 – Healthcare Devices (Devices with specific requirements for enhanced security and privacy)

3715 – Industrial Devices (Devices with advanced management, security and attestation requirements)

3716 **14.8.3.4.3 Requirements for Certification at Security Profile Black (Normative)**

3717 Every device with "currentprofile" Property of the "/oic/sec/sp" Resource designating a Security  
3718 Profile of "sp-black-v0", as defined in clause 14.8.2, must support each of the following:

- 3719 – Onboarding via OCF Rooted Certificate Chain, including PKI chain validation
- 3720 – Support for AES 128 encryption for data at rest and in transit.
- 3721 – Hardening minimums: manufacturer assertion of secure credential storage
- 3722 – In – in enumerated item #10 "The "/oic/sec/cred" Resource should contain credential(s) if  
3723 required by the selected OTM" is changed to require the credential be stored: "The  
3724 "/oic/sec/cred" Resource shall contain credential(s)."
- 3725 – The OCF Device shall include an X.509v3 OCF Compliance Extension (clause 9.4.2.2.4) in its  
3726 certificate and the extension's 'securityProfile' field shall contain sp-black-v0 represented by  
3727 the ocfSecurityProfileOID string, "1.3.6.1.4.1.51414.0.0.2.0".

3728 When a device supports the black profile, the "supportedprofiles" Property shall contain sp-black-  
3729 v0, represented by the OID string "1.3.6.1.4.1.51414.0.0.2.0", and may contain other profiles.

3730 When a manufacturer makes sp-black-v0 the default, by setting the "currentprofile" Property to  
3731 "1.3.6.1.4.1.51414.0.0.2.0", the "supportedprofiles" Property shall contain sp-black-v0.

3732 The OCF Rooted Certificate Chain and PKI Is defined by and structured within a framework  
3733 described in the supporting documents:

- 3734 – Certificate Profile (See 9.4.2)
- 3735 – Certificate Policy (see Certificate Policy document:  
3736 <https://openconnectivity.org/specs/OCF%20Certificate%20Policy.pdf>)

3737 **14.8.3.5 Security Profile Blue v0 (sp-blue-v0)**

3738 **14.8.3.5.1 Blue Profile General**

3739 The Security Profile Blue is used when manufacturers issue platform certificates for platforms  
3740 containing manufacturer-embedded keys. Compatibility with interoperable trusted platforms is  
3741 anticipated using certificate extensions defined by the Trusted Computing Group (TCG). OCF  
3742 Security Domain owners evaluate manufacturer supplied certificates and attributed data to  
3743 determine an appropriate OCF Security Profile that is configured for OCF Devices at onboarding.  
3744 OCF Devices may satisfy multiple OCF Security Profiles. The OCF Security Domain owner may  
3745 configure deployments using the Security Profile as OCF Security Domain partitioning criteria.

3746 Certificates issued to Blue Profile Devices shall be issued by a CA conforming to the CA Vetting  
3747 Criteria defined by OCF.

3748 **14.8.3.5.2 Platforms and Devices for Security Profile Blue v0**

3749 The OCF Security Profile Blue anticipates an ecosystem where platform vendors may differ from  
3750 OCF Device vendor and where platform vendors may implement trusted platforms that may conform  
3751 to industry standards defining trusted platforms. The OCF Security Profile Blue specifies  
3752 mechanisms for linking platforms with OCF Device(s) and for referencing quality assurance criteria  
3753 produced by OCF conformance operations. The OCF Security Domain owner evaluates these data  
3754 when an OCF Device is onboarded into the OCF Security Domain. Based on this evaluation the  
3755 OCF Security Domain owner determines which Security Profile may be applied during OCF Device  
3756 operation. All OCF Device types may be considered for evaluation using the OCF Security Profile  
3757 Blue.

3758 **14.8.3.5.3 Requirements for Certification at Security Profile Blue v0**

3759 The OCF Device satisfies the Blue profile v0 (sp-blue-v0) when all of the security normative for this  
3760 document version are satisfied and the following additional criteria are satisfied.

3761 OCF Blue profile defines the following OCF Device quality assurances:

- 3762 – The OCF Conformance criteria shall require vendor attestation that the conformant OCF Device  
3763 was hosted on one or more platforms that satisfies OCF Blue platform security assurances and  
3764 platform security and privacy functionality requirements.
- 3765 – The OCF Device achieving OCF Blue Security Profile compliance will be registered by OCF and  
3766 published by OCF in a machine readable format.
- 3767 – The OCF Blue Security Profile compliance registry may be digitally signed by an OCF owned  
3768 signing key.
- 3769 – The OCF Device shall include an X.509v3 OCF Compliance Extension (clause 9.4.2.2.4) in its  
3770 certificate and the extension's 'securityProfile' field shall contain sp-blue-v0 represented by the  
3771 ocfSecurityProfileOID string, "1.3.6.1.4.1.51414.0.0.3.0".
- 3772 – The OCF Device shall include an X.509v3 OCF CPL Attributes Extension (clause 9.4.2.2.7) in  
3773 its certificate.
- 3774 – The DOTS is expected to perform a lookup of the certification status of the OCF Device using  
3775 the OCF CPL Attributes Extension values and verify that the sp-blue-v0 OID is listed in the  
3776 extension's "securityprofiles" field.

3777 OCF Blue profile defines the following OCF Device security functionality:

- 3778 – OCF Device(s) shall be hosted on a platform where a cryptographic and secure storage  
3779 functions are hardened by the platform.
- 3780 – OCF Device(s) hosted on a platform shall expose accompanying manufacturer credentials using  
3781 the "/oic/sec/cred" Resource where the "credusage" Property contains the value  
3782 "oic.sec.cred.mfgcert".
- 3783 – OCF Device(s) that are hosted on a TCG-defined trusted platform should use an IEEE802.1AR  
3784 IDevID and should verify the "TCG Endorsement Key Credential". All TCG-defined  
3785 manufacturer credentials may be identified by the "oic.sec.cred.mfgcert" value of the  
3786 "credusage" Property of the "/oic/sec/cred" Resource. They may be used in response to  
3787 selection of the "oic.sec.doxm.mfgcert" owner transfer method.
- 3788 – OCF Device(s) shall use AES128 equivalent minimum protection for transmitted data. (See  
3789 NIST SP 800-57).
- 3790 – OCF Device(s) shall use AES128 equivalent minimum protection for stored data. (See NIST SP  
3791 800-57).
- 3792 – OCF Device(s) should use AES256 equivalent minimum protection for stored data. (See NIST  
3793 SP 800-57).
- 3794 – OCF Device(s) should protect the "/oic/sec/cred" resource using the platform provided secure  
3795 storage.
- 3796 – OCF Device(s) shall protect trust anchors (aka policy defining trusted CAs and pinned  
3797 certificates) using platform provided secure storage.
- 3798 – OCF Device(s) should check certificate revocation status for locally issued certificates.
- 3799 – The DOTS is expected to check certificate revocation status for all certificates in manufacturer  
3800 certificate path(s) if available. If a certificate is revoked, certificate validation fails and the  
3801 connection is refused. The DOTS may disregard revocation status results if unavailable.

3802 OCF Blue profile defines the following platform security assurances:

- 3803 – Platforms implementing cryptographic service provider (CSP) functionality and secure storage  
3804 functionality should be evaluated with a minimum FIPS140-2 Level 2 or Common Criteria EAL  
3805 Level 2.

- 3806 – Platforms implementing trusted platform functionality should be evaluated with a minimum  
3807 Common Criteria EAL Level 1.
- 3808 OCF Blue profile defines the following platform security and privacy functionality:
- 3809 – The Platform shall implement cryptographic service provider (CSP) functionality.
  - 3810 – Platform CSP functionality shall include cryptographic algorithms, random number generation,  
3811 secure time.
  - 3812 – The Platform shall implement AES128 equivalent protection for transmitted data. (See NIST SP  
3813 800-57).
  - 3814 – The Platform shall implement AES128 and AES256 equivalent protection for stored data. (See  
3815 NIST SP 800-57).
  - 3816 – Platforms hosting OCF Device(s) should implement a platform identifier following IEEE802.1AR  
3817 or Trusted Computing Group(TCG) specifications.
  - 3818 – Platforms based on Trusted Computing Group (TCG) platform definition that host OCF Device(s)  
3819 should supply TCG-defined manufacture certificates; also known as "TCG Endorsement Key  
3820 Credential" (which complies with IETF RFC 5280) and "TCG Platform Credential" (which  
3821 complies with IETF RFC 5755).
- 3822 When a device supports the blue profile, the "supportedprofiles" Property shall contain sp-blue-v0,  
3823 represented by the OID string "1.3.6.1.4.1.51414.0.0.3.0", and may contain other profiles.
- 3824 When a manufacturer makes sp-blue-v0 the default, by setting the "currentprofile" Property to  
3825 "1.3.6.1.4.1.51414.0.0.3.0", the "supportedprofiles" Property shall contain sp-blue-v0.
- 3826 During onboarding, while the device state is RFOTM, the DOTS may update the "currentprofile"  
3827 Property to one of the other values found in the "supportedprofiles" Property.
- 3828 **14.8.3.6 Security Profile Purple v0 (sp-purple-v0)**
- 3829 Every device with the "/oic/sec/sp" Resource designating "sp-purple-v0", as defined in clause  
3830 14.8.2 must support following minimum requirements
- 3831 – Hardening minimums: secure credential storage, software integrity validation, secure update.
  - 3832 – If a Certificate is used, the OCF Device shall include an X.509v3 OCF Compliance Extension  
3833 (clause 9.4.2.2.4) in its certificate and the extension's 'securityProfile' field shall contain sp-  
3834 purple-v0 represented by the ocfSecurityProfileOID string, "1.3.6.1.4.1.51414.0.0.4.0"
  - 3835 – The OCF Device shall include a X.509v3 OCFPLAttributes Extension (clause 9.4.2.2.7) in its  
3836 End-Entity Certificate when manufacturer certificate is used.
- 3837 Security Profile Purple has following optional security hardening requirements that the device can  
3838 additionally support.
- 3839 – Hardening additions: secure boot, hardware backed secure storage
  - 3840 – The OCF Device shall include a X.509v3 OCFSecurityClaims Extension (clause 9.4.2.2.6) in its  
3841 End-Entity Certificate and it shall include corresponding OIDs to the hardening additions  
3842 implemented and attested by the vendor. If there is no additional support for hardening  
3843 requirements, X.509v3 OCFSecurityClaims Extension shall be omitted.
- 3844 For software integrity validation, OCF Device(s) shall provide the integrity validation mechanism  
3845 for security critical executables such as cryptographic modules or secure service applications, and  
3846 they should be validated before the execution. The key used for validating the integrity must be  
3847 pinned at the least to the validating software module.
- 3848 For secure update, OCF Device(s) shall be able to update its firmware in a secure manner.

3849 For secure boot, OCF Device(s) shall implement the BIOS code (first-stage bootloader on ROM) to  
3850 be executed by the processor on power-on, and secure boot parameters to be provisioned by  
3851 tamper-proof memory. Also OCF Device(s) shall provide software module authentication for the  
3852 security critical executables and stop the boot process if any integrity of them is compromised.

3853 For hardware backed secure storage, OCF Device(s) shall store sensitive data in non-volatile  
3854 memory ("NVRAM") and prevent the retrieval of sensitive data through physical and/or electronic  
3855 attacks.

3856 More details on security hardening guidelines for software integrity validation, secure boot, secure  
3857 update, and hardware backed secure storage are described in 14.3, 14.5 and 14.2.2.2.

3858 Certificates issued to Purple Profile Devices shall be issued by a CA conforming to the CA Vetting  
3859 Criteria defined by OCF.

3860 When a device supports the purple profile, the "supportedprofiles" Property shall contain sp-purple-  
3861 v0, represented by the OID string "1.3.6.1.4.1.51414.0.0.4.0", and may contain other profiles.

3862 When a manufacturer makes sp-purple-v0 the default, by setting the "currentprofile" Property to  
3863 "1.3.6.1.4.1.51414.0.0.4.0", the "supportedprofiles" Property shall contain sp-purple-v0.

3864 **15 Device Type Specific Requirements**

3865 **15.1 Bridging Security**

3866 **15.1.1 Universal Requirements for Bridging to another Ecosystem**

3867 The Bridge shall go through OCF ownership transfer as any other onboarder would.

3868 The software of a Bridge shall be field updatable. (This requirement need not be tested but can be  
3869 certified via a vendor declaration.)

3870 Each VOD shall be onboarded by an OCF OBT. Each Virtual Bridged Device should be provisioned  
3871 as appropriate in the Bridged Protocol. In other words, VODs and Virtual Bridged Devices are  
3872 treated the same way as physical Devices. They are entities that have to be provisioned in their  
3873 network.

3874 Each VOD shall implement the behaviour required by ISO/IEC 30118-1:2018 and this document.  
3875 Each VOD shall perform authentication, access control, and encryption according to the security  
3876 settings it received from the OCF OBT. Each Virtual Bridged Device shall implement the security  
3877 requirements of the Bridged Protocol.

3878 In addition, in order to be considered secure from an OCF perspective, the Bridge Platform shall  
3879 use appropriate ecosystem-specific security options for communication between the Virtual Bridged  
3880 Devices instantiated by the Bridge and Bridged Devices. This security shall include mutual  
3881 authentication, and encryption and integrity protection of messages in the bridged ecosystem.

3882 A VOD may authenticate itself to the DOTS using the Manufacturer Certificate Based OTM (see  
3883 clause 7.3.6) with the Manufacturer Certificate and corresponding private key of the Bridge which  
3884 instantiated that VOD.

3885 A VOD may authenticate itself to the OCF Cloud using the Manufacturer Certificate and  
3886 corresponding private key of the Bridge which instantiated that VOD.

3887 A Bridge and the VODs created by that Bridge shall operate as independent Devices, with the  
3888 following exceptions:

3889 – If a Bridge creates a VOD while the Bridge is in an Unowned State, then the VOD shall be  
3890 created in an Unowned State.

3891 – An Unowned VOD shall not accept DTLS connection attempts nor TLS connection attempts nor  
3892 any other requests, including discovery requests, while the Bridge (that created that VOD) is  
3893 Unowned.

3894 – At any time when a Bridge is transitioning from Owned to Unowned State, all Unowned VODs  
3895 (created by that Bridge prior to the transition) shall drop any existing TLS and/or DTLS  
3896 connections.

3897 – At any time when a Bridge is transitioning from Unowned to Owned State, the Bridge shall  
3898 trigger all Unowned VODs (created by that Bridge prior to the transition) to become accessible  
3899 in RFOTM state, with internal state as if the VOD has just transitioned from RESET to RFOTM.

3900 – If a Bridge creates a VOD while the Bridge is in an Owned State, then the VOD shall become  
3901 accessible in RFOTM state, with internal state as if the VOD has just transitioned from RESET  
3902 to RFOTM.

3903 Table 53 intends to clarify this behaviour.

3904  
3905

**Table 53 – Dependencies of VOD Behaviour on Bridge state, as clarification of accompanying text**

Bridge state	Additional dependencies on VOD behaviour	
	VOD is Unowned (either just created, or created previously)	VOD is Owned
From unboxing Bridge until just prior to the end of transition of Bridge from Unowned to Owned	No accepting DTLS connection attempts nor TLS connection attempts nor any other requests, including discovery requests	Not applicable
At end of transition from Unowned to Owned	VOD becomes accessible in RFOTM following Bridge's transition. Internal state as if just transitioned from RESET.	As per normal Device
Owned	As per normal Device	As per normal Device
At Start of transition from Owned to Unowned	Drop any established TLS/DTLS connections, even if already partway through Device ownership	As per normal Device
Start of transition from Owned to Unowned, until just prior to the end of transition from Unowned to Owned.	No accepting DTLS connection attempts nor TLS connection attempts nor any other requests, including discovery requests	As per normal Device

3906 The "vods" Property of the "oic.r.vodlist" Resource on a Bridge reflects the details of all currently  
 3907 Owned VODs which have been created by that Bridge since the most recent hardware reset (if any)  
 3908 of the Bridge Platform (which removes all the created VODs), regardless of whether the VODs have  
 3909 the same owner as the Bridge or not. The entries in the "vods" Property are added and removed  
 3910 according to the following criteria:

- 3911 – Whenever a VOD created by a Bridge transitions from being Unowned to being Owned, then  
 3912 an entry for that VOD shall be added to the "vods" Property of the "oic.r.vodlist" Resource of  
 3913 that Bridge.
- 3914 – Whenever a VOD created by a Bridge transitions from being Owned to being Unowned, then  
 3915 entry for that VOD shall be removed from the "vods" Property of the "oic.r.vodlist" Resource of  
 3916 that Bridge. If that Bridge is currently in Unowned state, then the "oic.r.vodlist" Resource is not  
 3917 accessible, and the entry for that VOD shall be removed from the "vods" Property before or  
 3918 during the transition of that Bridge to the Owned state.
- 3919 – All other modifications of the list are not allowed.

3920 A Bridge shall only expose a secure OCF Endpoint for the "oic.r.vodlist" Resource.

3921 **15.1.2 Additional Security Requirements specific to Bridged Protocols**

3922 **15.1.2.1 Additional Security Requirements specific to the AllJoyn Protocol**

3923 For AllJoyn translator, an authenticated and authorized Client shall be able to block the  
 3924 communication of all OCF Devices with all Bridged Devices that don't communicate securely with  
 3925 the Bridge, by using the Bridge Device's "oic.r.securemode" Resource specified in ISO/IEC 30118-  
 3926 3:2018

3927 **15.1.2.2 Additional Security Requirements specific to the Bluetooth LE Protocol**

3928 A Bridge shall block the communication of all OCF Devices with all Bridged Devices that don't  
 3929 communicate securely with the Bridge.

3930 **15.1.2.3 Additional Security Requirements specific to the oneM2M Protocols**

3931 The Bridge shall implement oneM2M application access control as defined in the oneM2M Release  
 3932 3 Specifications.

3933 An Bridge shall block the communication of all OCF Devices with all Bridged Devices that don't  
 3934 communicate securely with the Bridge.

3935 **15.1.2.4 Additional Security Requirements specific to the U+ Protocol**

3936 A Bridge shall block the communication of all OCF Devices with all Bridged Devices that don't  
3937 communicate securely with the Bridge.

3938 **15.1.2.5 Additional Security Requirements specific to the Z-Wave Protocol**

3939 A Bridge shall block the communication of all OCF Devices with all Bridged Devices that don't  
3940 communicate securely with the Bridge.

3941 **15.1.2.6 Additional Security Requirements specific to the Zigbee Protocol**

3942 A Bridge shall block the communication of all OCF Devices with all Bridged Devices that don't  
3943 communicate securely with the Bridge.

3944 **15.1.2.7 Additional Security Requirements specific to the EnOcean Radio Protocol**

3945 A Bridge shall block the communication of all OCF Devices with all Bridged Devices that don't  
3946 communicate securely with the Bridge.

3947

3948

3949

3950

3951

3952

3953

3954

3955

3956

3957

3958

3959

3960

3961

3962

3963

3964

3965

3966

3967 .



3968  
3969  
3970

## Annex A (informative) Access Control Examples

3971

### Example OCF ACL Resource

3972 Figure A-1 shows how a "/oic/sec/acl2" Resource could be configured to enforce an example  
3973 access policy on the Server.

```
3974 {
3975   "aclist2": [
3976     {
3977       // Subject with ID ...01 should access two named Resources with access mode "CRUDN" (Create, Retrieve, Update,
3978       Delete and Notify)
3979       "subject": {"uuid": "XXX-...-XX01"},
3980       "resources": [
3981         {"href": "/oic/sh/light/1"},
3982         {"href": "/oic/sh/temp/0"}
3983       ],
3984       "permission": 31, // 31 dec = 0b0001 1111 which maps to ---N DURC
3985       "validity": [
3986         // The period starting at 18:00:00 UTC, on January 1, 2015 and
3987         // ending at 07:00:00 UTC on January 2, 2015
3988         "period": ["20150101T180000Z/20150102T070000Z"],
3989         // Repeats the {period} every week until the last day of Jan. 2015.
3990         "recurrence": ["RRULE:FREQ=WEEKLY;UNTIL=20150131T070000Z"]
3991       ],
3992       "aceid": 1
3993     }
3994   ],
3995   // An ACL provisioning and management service should be identified as
3996   // the resource owner
3997   "rowneruid": "0685B960-736F-46F7-BEC0-9E6CBD61ADC1"
3998 }
3999
```

**Figure A-1 – Example "/oic/sec/acl2" Resource**

4000  
4001  
4002

## Annex B (Informative) Execution Environment Security Profiles

4003 Given that IoT verticals and Devices will not be of uniform capabilities, a one-size-fits all security  
4004 robustness requirements meeting all IOT applications and services will not serve the needs of OCF,  
4005 and security profiles of varying degree of robustness (trustworthiness), cost and complexity have  
4006 to be defined. To address a large ecosystem of vendors, the profiles can only be defined as  
4007 requirements and the exact solutions meeting those requirements are specific to the vendors' open  
4008 or proprietary implementations, and thus in most part outside scope of this document.

4009 To align with the rest of OCF documents, where Device classifications follow IETF RFC 7228  
4010 (Terminology for constrained node networks) methodology, we limit the number of security profiles  
4011 to a maximum of 3 (see Table B.1). However, our understanding is OCF capabilities criteria for  
4012 each of 3 classes will be more fit to the current IoT chip market than that of IETF.

4013 Given the extremely low level of resources at class 0, our expectation is that class 0 Devices are  
4014 either capable of no security functionality or easily breakable security that depend on environmental  
4015 (e.g. availability of human) factors to perform security functions. This means the class 0 will not be  
4016 equipped with an SEE.

4017

**Table B.1 – OCF Security Profile**

Platform class	SEE	Robustness level
0	No	N/A
1	Yes	Low
2	Yes	High

4018 NOTE This analysis acknowledges that these Platform classifications do not take into consideration of possibility of  
4019 security co-processor or other hardware security capability that augments classification criteria (namely CPU speed,  
4020 memory, storage).

4021  
4022  
4023

## Annex C (normative) Resource Type definitions

### 4024 C.1 List of Resource Type definitions

4025 Table C.1 contains the list of defined security resources in this document.

4026 **Table C.1 – Alphabetized list of security resources**

Friendly Name (informative)	Resource Type (rt)	Clause
Access Control List 2	oic.r.acl2	C.2
Certificate Signing Request	oic.r.csr	C.4
Credential	oic.r.cred	C.3
Device owner transfer method	oic.r.doxm	C.5
Device Provisioning Status	oic.r.pstat	C.6
Roles	oic.r.roles	C.7
Security Profile	oic.r.sp	C.8
Account	oic.r.account	Moved to OCF Cloud Security document
Account Session	oic.r.session	Moved to OCF Cloud Security document
Account Token Refresh	oic.r.tokenrefresh	Moved to OCF Cloud Security document

### 4027 C.2 Access Control List-2

#### 4028 C.2.1 Introduction

4029 This Resource specifies the local access control list.  
4030 When used without query parameters, all the ACE entries are returned.  
4031 When used with a query parameter, only the ACEs matching the specified  
4032 parameter are returned.  
4033

#### 4034 C.2.2 Well-known URI

4035 /oic/sec/acl2

#### 4036 C.2.3 Resource type

4037 The Resource Type is defined as: "oic.r.acl2".

#### 4038 C.2.4 OpenAPI 2.0 definition

```
4039 {  
4040   "swagger": "2.0",  
4041   "info": {  
4042     "title": "Access Control List-2",  
4043     "version": "20190111",  
4044     "license": {  
4045       "name": "OCF Data Model License",  
4046       "url":  
4047       "https://github.com/openconnectivityfoundation/core/blob/e28a9e0a92e17042ba3e83661e4c0fbce8bdc4ba/LI  
4048       CENSE.md",  
4049       "x-copyright": "copyright 2016-2017, 2019 Open Connectivity Foundation, Inc. All rights  
4050       reserved."  
4051     },  
4052   }
```

```

4052     "termsOfService": "https://openconnectivityfoundation.github.io/core/DISCLAIMER.md"
4053   },
4054   "schemes": ["http"],
4055   "consumes": ["application/json"],
4056   "produces": ["application/json"],
4057   "paths": {
4058     "/oic/sec/acl2" : {
4059       "get": {
4060         "description": "This Resource specifies the local access control list.\nWhen used without
4061 query parameters, all the ACE entries are returned.\nWhen used with a query parameter, only the ACEs
4062 matching the specified\nparameter are returned.\n",
4063         "parameters": [
4064           {"$ref": "#/parameters/interface"},
4065           {"$ref": "#/parameters/ace-filtered"}
4066         ],
4067         "responses": {
4068           "200": {
4069             "description": "",
4070             "x-example":
4071               {
4072                 "rt" : ["oic.r.acl2"],
4073                 "aclist2": [
4074                   {
4075                     "aceid": 1,
4076                     "subject": {
4077                       "authority": "484b8a51-cb23-46c0-a5f1-b4aebef50ebe",
4078                       "role": "SOME_STRING"
4079                     },
4080                     "resources": [
4081                       {
4082                         "href": "/light"
4083                       },
4084                       {
4085                         "href": "/door"
4086                       }
4087                     ],
4088                     "permission": 24
4089                   },
4090                   {
4091                     "aceid": 2,
4092                     "subject": {
4093                       "uuid": "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9"
4094                     },
4095                     "resources": [
4096                       {
4097                         "href": "/light"
4098                       },
4099                       {
4100                         "href": "/door"
4101                       }
4102                     ],
4103                     "permission": 24
4104                   },
4105                   {
4106                     "aceid": 3,
4107                     "subject": {"conntype": "anon-clear"},
4108                     "resources": [
4109                       {
4110                         "href": "/light"
4111                       },
4112                       {
4113                         "href": "/door"
4114                       }
4115                     ],
4116                     "permission": 16,
4117                     "validity": [
4118                       {
4119                         "period": "20160101T180000Z/20170102T070000Z",
4120                         "recurrence": [ "DSTART:XXXXX",
4121 "RRULE:FREQ=DAILY;UNTIL=20180131T140000Z;BYMONTH=1" ]
4122                       },

```

```

4123         {
4124             "period": "20160101T180000Z/PT5H30M",
4125             "recurrence": [ "RRULE:FREQ=DAILY;UNTIL=20180131T140000Z;BYMONTH=1" ]
4126         }
4127     ]
4128 }
4129 ],
4130     "rowneruuid": "de305d54-75b4-431b-adb2-eb6b9e546014"
4131 },
4132     "schema": { "$ref": "#/definitions/Acl2" }
4133 },
4134     "400": {
4135         "description": "The request is invalid."
4136     }
4137 }
4138 },
4139     "post": {
4140         "description": "Updates the ACL Resource with the provided ACEs.\n\nACEs provided in the
4141 update with aceids not currently in the ACL\nResource are added.\n\nACEs provided in the update with
4142 aceid(s) already in the ACL completely\nreplace the ACE(s) in the ACL Resource.\n\nACEs provided in
4143 the update without aceid properties are added and\nassigned unique aceids in the ACL Resource.\n\n",
4144         "parameters": [
4145             { "$ref": "#/parameters/interface" },
4146             { "$ref": "#/parameters/ace-filtered" },
4147             {
4148                 "name": "body",
4149                 "in": "body",
4150                 "required": true,
4151                 "schema": { "$ref": "#/definitions/Acl2-Update" },
4152                 "x-example":
4153                     {
4154                         "aclist2": [
4155                             {
4156                                 "aceid": 1,
4157                                 "subject": {
4158                                     "authority": "484b8a51-cb23-46c0-a5f1-b4aebef50ebe",
4159                                     "role": "SOME_STRING"
4160                                 },
4161                                 "resources": [
4162                                     {
4163                                         "href": "/light"
4164                                     },
4165                                     {
4166                                         "href": "/door"
4167                                     }
4168                                 ],
4169                                 "permission": 24
4170                             },
4171                             {
4172                                 "aceid": 3,
4173                                 "subject": {
4174                                     "uuid": "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9"
4175                                 },
4176                                 "resources": [
4177                                     {
4178                                         "href": "/light"
4179                                     },
4180                                     {
4181                                         "href": "/door"
4182                                     }
4183                                 ],
4184                                 "permission": 24
4185                             }
4186                         ],
4187                         "rowneruuid": "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9"
4188                     }
4189             },
4190         ],
4191         "responses": {
4192             "400": {
4193                 "description": "The request is invalid."

```

```

4194         },
4195         "201": {
4196             "description": "The ACL entry is created."
4197         },
4198         "204": {
4199             "description": "The ACL entry is updated."
4200         }
4201     }
4202 },
4203     "delete": {
4204         "description": "Deletes ACL entries.\nWhen DELETE is used without query parameters, all the
4205 ACE entries are deleted.\nWhen DELETE is used with a query parameter, only the ACEs matching
4206 the\nspecified parameter are deleted.\n",
4207         "parameters": [
4208             {"$ref": "#/parameters/interface"},
4209             {"$ref": "#/parameters/ace-filtered"}
4210         ],
4211         "responses": {
4212             "200": {
4213                 "description": "The matching ACEs or the entire ACL Resource has been successfully
4214 deleted."
4215             },
4216             "400": {
4217                 "description": "The request is invalid."
4218             }
4219         }
4220     }
4221 },
4222 },
4223     "parameters": {
4224         "interface": {
4225             "in": "query",
4226             "name": "if",
4227             "type": "string",
4228             "enum": ["oic.if.baseline"]
4229         },
4230         "ace-filtered": {
4231             "in": "query",
4232             "name": "aceid",
4233             "required": false,
4234             "type": "integer",
4235             "description": "Only applies to the ACE with the specified aceid.",
4236             "x-example": 2112
4237         }
4238     },
4239     "definitions": {
4240         "Acl2": {
4241             "properties": {
4242                 "owneruuid": {
4243                     "description": "The value identifies the unique Resource owner\nFormat pattern according
4244 to IETF RFC 4122.",
4245                     "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-
4246 9]{12}$",
4247                     "type": "string"
4248                 },
4249                 "rt": {
4250                     "description": "Resource Type of the Resource.",
4251                     "items": {
4252                         "maxLength": 64,
4253                         "type": "string",
4254                         "enum": ["oic.r.acl2"]
4255                     },
4256                     "minItems": 1,
4257                     "maxItems": 1,
4258                     "readOnly": true,
4259                     "type": "array"
4260                 },
4261                 "aclist2": {
4262                     "description": "Access Control Entries in the ACL Resource.",
4263                     "items": {
4264                         "properties": {

```

```

4265         "aceid": {
4266             "description": "An identifier for the ACE that is unique within the ACL. In cases
4267 where it isn't supplied in an update, the Server will add the ACE and assign it a unique value.",
4268             "minimum": 1,
4269             "type": "integer"
4270         },
4271         "permission": {
4272             "description": "Bitmask encoding of CRUDN permission\nThe encoded bitmask indicating
4273 permissions.",
4274             "x-detail-desc": [
4275                 "0 - No permissions",
4276                 "1 - Create permission is granted",
4277                 "2 - Read, observe, discover permission is granted",
4278                 "4 - Write, update permission is granted",
4279                 "8 - Delete permission is granted",
4280                 "16 - Notify permission is granted"
4281             ],
4282             "maximum": 31,
4283             "minimum": 0,
4284             "type": "integer"
4285         },
4286         "resources": {
4287             "description": "References the application's Resources to which a security policy
4288 applies.",
4289             "items": {
4290                 "description": "Each Resource must have at least one of these properties set.",
4291                 "properties": {
4292                     "href": {
4293                         "description": "When present, the ACE only applies when the href matches\nThis
4294 is the target URI, it can be specified as a Relative Reference or fully-qualified URI.",
4295                         "format": "uri",
4296                         "maxLength": 256,
4297                         "type": "string"
4298                     },
4299                     "wc": {
4300                         "description": "A wildcard matching policy.",
4301                         "pattern": "^[~+*]$",
4302                         "type": "string"
4303                     }
4304                 },
4305                 "type": "object"
4306             },
4307             "type": "array"
4308         },
4309         "subject": {
4310             "anyOf": [
4311                 {
4312                     "description": "This is the Device identifier.",
4313                     "properties": {
4314                         "uuid": {
4315                             "description": "A UUID Device ID\nFormat pattern according to IETF RFC
4316 4122.",
4317                             "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-
4318 fA-F0-9]{12}$",
4319                             "type": "string"
4320                         }
4321                     },
4322                     "required": [
4323                         "uuid"
4324                     ],
4325                     "type": "object"
4326                 },
4327                 {
4328                     "description": "Security role specified as an <Authority> & <Rolename>. A NULL
4329 <Authority> refers to the local entity or Device.",
4330                     "properties": {
4331                         "authority": {
4332                             "description": "The Authority component of the entity being identified. A
4333 NULL <Authority> refers to the local entity or Device.",
4334                             "type": "string"
4335                         }

```

```

4336         "role": {
4337             "description": "The ID of the role being identified.",
4338             "type": "string"
4339         }
4340     },
4341     "required": [
4342         "role"
4343     ],
4344     "type": "object"
4345 },
4346 {
4347     "properties": {
4348         "conntype": {
4349             "description": "This property allows an ACE to be matched based on the
4350 connection or message type.",
4351             "x-detail-desc": [
4352                 "auth-crypt - ACE applies if the Client is authenticated and the data
4353 channel or message is encrypted and integrity protected",
4354                 "anon-clear - ACE applies if the Client is not authenticated and the data
4355 channel or message is not encrypted but may be integrity protected"
4356             ],
4357             "enum": [
4358                 "auth-crypt",
4359                 "anon-clear"
4360             ],
4361             "type": "string"
4362         }
4363     },
4364     "required": [
4365         "conntype"
4366     ],
4367     "type": "object"
4368 }
4369 ]
4370 },
4371 "validity": {
4372     "description": "validity is an array of time-pattern objects.",
4373     "items": {
4374         "description": "The time-pattern contains a period and recurrence expressed in
4375 RFC5545 syntax.",
4376         "properties": {
4377             "period": {
4378                 "description": "String represents a period using the RFC5545 Period.",
4379                 "type": "string"
4380             },
4381             "recurrence": {
4382                 "description": "String array represents a recurrence rule using the RFC5545
4383 Recurrence.",
4384                 "items": {
4385                     "type": "string"
4386                 },
4387                 "type": "array"
4388             }
4389         },
4390         "required": [
4391             "period"
4392         ],
4393         "type": "object"
4394     },
4395     "type": "array"
4396 }
4397 },
4398 "required": [
4399     "aceid",
4400     "resources",
4401     "permission",
4402     "subject"
4403 ],
4404 "type": "object"
4405 },
4406 "type": "array"

```



```

4407     },
4408     "n": {
4409       "$ref":
4410       "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
4411       schema.json#/definitions/n"
4412     },
4413     "id": {
4414       "$ref":
4415       "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
4416       schema.json#/definitions/id"
4417     },
4418     "if" : {
4419       "description": "The interface set supported by this Resource.",
4420       "items": {
4421         "enum": [
4422           "oic.if.baseline"
4423         ],
4424         "type": "string"
4425       },
4426       "minItems": 1,
4427       "maxItems": 1,
4428       "readOnly": true,
4429       "type": "array"
4430     }
4431   },
4432   "type" : "object",
4433   "required": ["aclist2", "rowneruuid"]
4434 },
4435 "Acl2-Update" : {
4436   "properties": {
4437     "rowneruuid" : {
4438       "description": "The value identifies the unique Resource owner\n Format pattern according
4439 to IETF RFC 4122.",
4440       "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-
4441 9]{12}$",
4442       "type": "string"
4443     },
4444     "aclist2" : {
4445       "description": "Access Control Entries in the ACL Resource.",
4446       "items": {
4447         "properties": {
4448           "aceid": {
4449             "description": "An identifier for the ACE that is unique within the ACL. In cases
4450 where it isn't applied in an update, the Server will add the ACE and assign it a unique value.",
4451             "minimum": 1,
4452             "type": "integer"
4453           },
4454           "permission": {
4455             "description": "Bitmask encoding of CRUDN permission\nThe encoded bitmask indicating
4456 permissions.",
4457             "x-detail-desc": [
4458               "0 - No permissions",
4459               "1 - Create permission is granted",
4460               "2 - Read, observe, discover permission is granted",
4461               "4 - Write, update permission is granted",
4462               "8 - Delete permission is granted",
4463               "16 - Notify permission is granted"
4464             ],
4465             "maximum": 31,
4466             "minimum": 0,
4467             "type": "integer"
4468           },
4469           "resources": {
4470             "description": "References the application's Resources to which a security policy
4471 applies.",
4472             "items": {
4473               "description": "Each Resource must have at least one of these properties set.",
4474               "properties": {
4475                 "href": {
4476                   "description": "When present, the ACE only applies when the href matches\nThis
4477 is the target URI, it can be specified as a Relative Reference or fully-qualified URI.",

```

```

4478         "format": "uri",
4479         "maxLength": 256,
4480         "type": "string"
4481     },
4482     "wc": {
4483         "description": "A wildcard matching policy.",
4484         "x-detail-desc": [
4485             "+ - Matches all discoverable Resources",
4486             "- - Matches all non-discoverable Resources",
4487             "* - Matches all Resources"
4488         ],
4489         "enum": [
4490             "+",
4491             "-",
4492             "*"
4493         ],
4494         "type": "string"
4495     }
4496 },
4497 "type": "object"
4498 },
4499 "type": "array"
4500 },
4501 "subject": {
4502     "anyOf": [
4503         {
4504             "description": "This is the Device identifier.",
4505             "properties": {
4506                 "uuid": {
4507                     "description": "A UUID Device ID\n Format pattern according to IETF RFC
4122.",
4508                     "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-
4509 fA-F0-9]{12}$",
4510                     "type": "string"
4511                 }
4512             },
4513             "required": [
4514                 "uuid"
4515             ],
4516             "type": "object"
4517         },
4518         {
4519             "description": "Security role specified as an <Authority> & <Rolename>. A NULL
<Authority> refers to the local entity or Device.",
4520             "properties": {
4521                 "authority": {
4522                     "description": "The Authority component of the entity being identified. A
4523 NULL <Authority> refers to the local entity or Device.",
4524                     "type": "string"
4525                 },
4526                 "role": {
4527                     "description": "The ID of the role being identified.",
4528                     "type": "string"
4529                 }
4530             },
4531             "required": [
4532                 "role"
4533             ],
4534             "type": "object"
4535         },
4536         {
4537             "properties": {
4538                 "conntype": {
4539                     "description": "This property allows an ACE to be matched based on the
4540 connection or message type.",
4541                     "x-detail-desc": [
4542                         "auth-crypt - ACE applies if the Client is authenticated and the data
4543 channel or message is encrypted and integrity protected",
4544                         "anon-clear - ACE applies if the Client is not authenticated and the data
4545 channel or message is not encrypted but may be integrity protected"
4546                     ],
4547                 }
4548             }
4549         }
4550     ]
4551 }

```

```

4549         "enum": [
4550             "auth-crypt",
4551             "anon-clear"
4552         ],
4553         "type": "string"
4554     }
4555 },
4556     "required": [
4557         "conntype"
4558     ],
4559     "type": "object"
4560 }
4561 ]
4562 },
4563 "validity": {
4564     "description": "validity is an array of time-pattern objects.",
4565     "items": {
4566         "description": "The time-pattern contains a period and recurrence expressed in
RFC5545 syntax.",
4567         "properties": {
4568             "period": {
4569                 "description": "String represents a period using the RFC5545 Period.",
4570                 "type": "string"
4571             },
4572             "recurrence": {
4573                 "description": "String array represents a recurrence rule using the RFC5545
Recurrence.",
4574                 "items": {
4575                     "type": "string"
4576                 },
4577                 "type": "array"
4578             }
4579         }
4580     },
4581     "required": [
4582         "period"
4583     ],
4584     "type": "object"
4585 },
4586     "type": "array"
4587 }
4588 },
4589 "required": [
4590     "resources",
4591     "permission",
4592     "subject"
4593 ],
4594     "type": "object"
4595 },
4596     "type": "array"
4597 }
4598 },
4599     "type": "object"
4600 }
4601 }
4602 }
4603 }
4604

```

4605 **C.2.5 Property definition**

4606 Table C-1 defines the Properties that are part of the "oic.r.acl2" Resource Type.

4607 **Table C-1 – The Property definitions of the Resource with type "rt" = "oic.r.acl2".**

Property name	Value type	Mandatory	Access mode	Description
rowneruuid	string	Yes	Read Write	The value identifies the unique Resource owner Format pattern

				according to IETF RFC 4122.
rt	array: see schema	No	Read Only	Resource Type of the Resource.
aclist2	array: see schema	Yes	Read Write	Access Control Entries in the ACL Resource.
n	multiple types: see schema	No	Read Write	
id	multiple types: see schema	No	Read Write	
if	array: see schema	No	Read Only	The interface set supported by this Resource.
rowneruuid	string	No	Read Write	The value identifies the unique Resource owner Format pattern according to IETF RFC 4122.
aclist2	array: see schema	No	Read Write	Access Control Entries in the ACL Resource.

4608 **C.2.6 CRUDN behaviour**

4609 Table C-2 defines the CRUDN operations that are supported on the "oic.r.acl2" Resource Type.

4610 **Table C-2 – The CRUDN operations of the Resource with type "rt" = "oic.r.acl2".**

Create	Read	Update	Delete	Notify
	get	post	delete	observe

4611 **C.3 Credential**

4612 **C.3.1 Introduction**

4613 This Resource specifies credentials a Device may use to establish secure communication.

4614 Retrieves the credential data.

4615 When used without query parameters, all the credential entries are returned.

4616 When used with a query parameter, only the credentials matching the specified parameter are returned.

4617

4618 Note that write-only credential data will not be returned.

4619

4620

4621 **C.3.2 Well-known URI**

4622 /oic/sec/cred

4623 **C.3.3 Resource type**

4624 The Resource Type is defined as: "oic.r.cred".

4625 **C.3.4 OpenAPI 2.0 definition**

4626 {  
4627 "swagger": "2.0",  
4628 "info": {

```

4629     "title": "Credential",
4630     "version": "v1.0-20181031",
4631     "license": {
4632         "name": "OCF Data Model License",
4633         "url":
4634         "https://github.com/openconnectivityfoundation/core/blob/e28a9e0a92e17042ba3e83661e4c0fbce8bdc4ba/LI
4635         CENSE.md",
4636         "x-copyright": "copyright 2016-2017, 2019 Open Connectivity Foundation, Inc. All rights
4637         reserved."
4638     },
4639     "termsOfService": "https://openconnectivityfoundation.github.io/core/DISCLAIMER.md"
4640 },
4641 "schemas": ["http"],
4642 "consumes": ["application/json"],
4643 "produces": ["application/json"],
4644 "paths": {
4645     "/oic/sec/cred" : {
4646         "get": {
4647             "description": "This Resource specifies credentials a Device may use to establish secure
4648             communication.\nRetrieves the credential data.\nWhen used without query parameters, all the
4649             credential entries are returned.\nWhen used with a query parameter, only the credentials matching
4650             the specified\nparameter are returned.\n\nNote that write-only credential data will not be
4651             returned.\n",
4652             "parameters": [
4653                 {"$ref": "#/parameters/interface"}
4654                 , {"$ref": "#/parameters/cred-filtered-credid"}
4655                 , {"$ref": "#/parameters/cred-filtered-subjectuuid"}
4656             ],
4657             "responses": {
4658                 "200": {
4659                     "description": "",
4660                     "x-example":
4661                     {
4662                         "rt": ["oic.r.cred"],
4663                         "creds": [
4664                             {
4665                                 "credid": 55,
4666                                 "subjectuuid": "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
4667                                 "roleid": {
4668                                     "authority": "484b8a51-cb23-46c0-a5f1-b4aebef50ebe",
4669                                     "role": "SOME_STRING"
4670                                 },
4671                                 "credtype": 32,
4672                                 "publicdata": {
4673                                     "encoding": "oic.sec.encoding.pem",
4674                                     "data": "PEM-ENCODED-VALUE"
4675                                 },
4676                                 "privatedata": {
4677                                     "encoding": "oic.sec.encoding.raw",
4678                                     "data": "RAW-ENCODED-VALUE",
4679                                     "handle": 4
4680                                 },
4681                                 "optionaldata": {
4682                                     "revstat": false,
4683                                     "encoding": "oic.sec.encoding.pem",
4684                                     "data": "PEM-ENCODED-VALUE"
4685                                 },
4686                                 "period": "20160101T180000Z/20170102T070000Z",
4687                                 "crms": [ "oic.sec.crm.pk10" ]
4688                             },
4689                             {
4690                                 "credid": 56,
4691                                 "subjectuuid": "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
4692                                 "roleid": {
4693                                     "authority": "484b8a51-cb23-46c0-a5f1-b4aebef50ebe",
4694                                     "role": "SOME_STRING"
4695                                 },
4696                                 "credtype": 1,
4697                                 "publicdata": {
4698                                     "encoding": "oic.sec.encoding.pem",
4699                                     "data": "PEM-ENCODED-VALUE"

```

```

4700         },
4701         "privatedata": {
4702             "encoding": "oic.sec.encoding.base64",
4703             "data": "BASE-64-ENCODED-VALUE",
4704             "handle": 4
4705         },
4706         "optionaldata": {
4707             "revstat": false,
4708             "encoding": "oic.sec.encoding.pem",
4709             "data": "PEM-ENCODED-VALUE"
4710         },
4711         "period": "20160101T180000Z/20170102T070000Z",
4712         "crms": [ "oic.sec.crm.pk10" ]
4713     }
4714 ],
4715     "xowneruuid": "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9"
4716 }
4717 ,
4718     "schema": { "$ref": "#/definitions/Cred" }
4719 },
4720 "400": {
4721     "description": "The request is invalid."
4722 }
4723 }
4724 },
4725 "post": {
4726     "description": "Updates the credential Resource with the provided
4727 credentials.\n\nCredentials provided in the update with credid(s) not currently in the\ncredential
4728 Resource are added.\n\nCredentials provided in the update with credid(s) already in the\ncredential
4729 Resource completely replace the creds in the credential\nResource.\n\nCredentials provided in the
4730 update without credid(s) properties are\nadded and assigned unique credid(s) in the credential
4731 Resource.\n",
4732     "parameters": [
4733         { "$ref": "#/parameters/interface" },
4734         {
4735             "name": "body",
4736             "in": "body",
4737             "required": true,
4738             "schema": { "$ref": "#/definitions/Cred-Update" },
4739             "x-example":
4740                 {
4741                     "creds": [
4742                         {
4743                             "credid": 55,
4744                             "subjectuuid": "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
4745                             "roleid": {
4746                                 "authority": "484b8a51-cb23-46c0-a5f1-b4aebef50ebe",
4747                                 "role": "SOME_STRING"
4748                             },
4749                             "credtype": 32,
4750                             "publicdata": {
4751                                 "encoding": "oic.sec.encoding.pem",
4752                                 "data": "PEM-ENCODED-VALUE"
4753                             },
4754                             "privatedata": {
4755                                 "encoding": "oic.sec.encoding.raw",
4756                                 "data": "RAW-ENCODED-VALUE",
4757                                 "handle": 4
4758                             },
4759                             "optionaldata": {
4760                                 "revstat": false,
4761                                 "encoding": "oic.sec.encoding.pem",
4762                                 "data": "PEM-ENCODED-VALUE"
4763                             },
4764                             "period": "20160101T180000Z/20170102T070000Z",
4765                             "crms": [ "oic.sec.crm.pk10" ]
4766                         },
4767                         {
4768                             "credid": 56,
4769                             "subjectuuid": "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
4770                             "roleid": {

```

```

4771         "authority": "484b8a51-cb23-46c0-a5f1-b4aebef50ebe",
4772         "role": "SOME_STRING"
4773     },
4774     "credtype": 1,
4775     "publicdata": {
4776         "encoding": "oic.sec.encoding.pem",
4777         "data": "PEM-ENCODED-VALUE"
4778     },
4779     "privatedata": {
4780         "encoding": "oic.sec.encoding.base64",
4781         "data": "BASE-64-ENCODED-VALUE",
4782         "handle": 4
4783     },
4784     "optionaldata": {
4785         "revstat": false,
4786         "encoding": "oic.sec.encoding.pem",
4787         "data": "PEM-ENCODED-VALUE"
4788     },
4789     "period": "20160101T180000Z/20170102T070000Z",
4790     "crms": [ "oic.sec.crm.pk10" ]
4791 }
4792 ],
4793 "rowneruid": "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9"
4794 }
4795 }
4796 ],
4797 "responses": {
4798     "400": {
4799         "description": "The request is invalid."
4800     },
4801     "201": {
4802         "description": "The credential entry is created."
4803     },
4804     "204": {
4805         "description": "The credential entry is updated."
4806     }
4807 }
4808 },
4809 "delete": {
4810     "description": "Deletes credential entries.\nWhen DELETE is used without query parameters,
4811 all the cred entries are deleted.\nWhen DELETE is used with a query parameter, only the entries
4812 matching\nthe query parameter are deleted.\n",
4813     "parameters": [
4814         {"$ref": "#/parameters/interface"},
4815         {"$ref": "#/parameters/cred-filtered-credid"},
4816         {"$ref": "#/parameters/cred-filtered-subjectuid"}
4817     ],
4818     "responses": {
4819         "400": {
4820             "description": "The request is invalid."
4821         },
4822         "204": {
4823             "description": "The specific credential(s) or the the entire credential Resource has
4824 been successfully deleted."
4825         }
4826     }
4827 }
4828 }
4829 },
4830 "parameters": {
4831     "interface": {
4832         "in": "query",
4833         "name": "if",
4834         "type": "string",
4835         "enum": ["oic.if.baseline"]
4836     },
4837     "cred-filtered-credid": {
4838         "in": "query",
4839         "name": "credid",
4840         "required": false,
4841         "type": "integer",

```

```

4842         "description" : "Only applies to the credential with the specified credid.",
4843         "x-example" : 2112
4844     },
4845     "cred-filtered-subjectuuid" : {
4846         "in" : "query",
4847         "name" : "subjectuuid",
4848         "required" : false,
4849         "type" : "string",
4850         "description" : "Only applies to credentials with the specified subject UUID.",
4851         "x-example" : "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9"
4852     }
4853 },
4854 "definitions": {
4855     "Cred" : {
4856         "properties": {
4857             "rowneruuid" : {
4858                 "description": "Format pattern according to IETF RFC 4122.",
4859                 "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{12}$",
4860                 "type": "string"
4861             },
4862         },
4863         "rt" : {
4864             "description": "Resource Type of the Resource.",
4865             "items": {
4866                 "maxLength": 64,
4867                 "type": "string",
4868                 "enum": ["oic.r.cred"]
4869             },
4870             "minItems": 1,
4871             "readOnly": true,
4872             "type": "array",
4873             "uniqueItems": true
4874         },
4875         "n": {
4876             "$ref":
4877 "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
4878 schema.json#/definitions/n"
4879         },
4880         "id": {
4881             "$ref":
4882 "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
4883 schema.json#/definitions/id"
4884         },
4885         "creds" : {
4886             "description": "List of credentials available at this Resource.",
4887             "items": {
4888                 "properties": {
4889                     "credid": {
4890                         "description": "Local reference to a credential Resource.",
4891                         "type": "integer"
4892                     },
4893                     "credtype": {
4894                         "description": "Representation of this credential's type\nCredential Types - Cred
4895 type encoded as a bitmask.0 - Empty credential used for testing1 - Symmetric pair-wise key2 -
4896 Symmetric group key4 - Asymmetric signing key8 - Asymmetric signing key with certificatel6 - PIN or
4897 password32 - Asymmetric encryption key.",
4898                         "maximum": 63,
4899                         "minimum": 0,
4900                         "type": "integer"
4901                     },
4902                     "credusage": {
4903                         "description": "A string that provides hints about how/where the cred is used\nThe
4904 type of credusage.oic.sec.cred.trustca - Trust certificateoic.sec.cred.cert -
4905 Certificateoic.sec.cred.rolecert - Role Certificateoic.sec.cred.mfgtrustca - Manufacturer
4906 Certificate Trust Anchoroic.sec.cred.mfgcert - Manufacturer Certificate.",
4907                         "enum": [
4908                             "oic.sec.cred.trustca",
4909                             "oic.sec.cred.cert",
4910                             "oic.sec.cred.rolecert",
4911                             "oic.sec.cred.mfgtrustca",
4912                             "oic.sec.cred.mfgcert"

```



```

4913     ],
4914     "type": "string"
4915   },
4916   "crms": {
4917     "description": "The refresh methods that may be used to update this credential.",
4918     "items": {
4919       "description": "Each enum represents a method by which the credentials are
4920 refreshed.oic.sec.crm.pro - Credentials refreshed by a provisioning serviceoic.sec.crm.rdp -
4921 Credentials refreshed by a key agreement protocol and random PINoic.sec.crm.psk - Credentials
4922 refreshed by a key agreement protocoloic.sec.crm.skdc - Credentials refreshed by a key distribution
4923 serviceoic.sec.crm.pk10 - Credentials refreshed by a PKCS#10 request to a CA.",
4924       "enum": [
4925         "oic.sec.crm.pro",
4926         "oic.sec.crm.psk",
4927         "oic.sec.crm.rdp",
4928         "oic.sec.crm.skdc",
4929         "oic.sec.crm.pk10"
4930       ],
4931       "type": "string"
4932     },
4933     "type": "array",
4934     "uniqueItems": true
4935   },
4936   "optionaldata": {
4937     "description": "Credential revocation status information\nOptional credential
4938 contents describes revocation status for this credential.",
4939     "properties": {
4940       "data": {
4941         "description": "The encoded structure.",
4942         "type": "string"
4943       },
4944       "encoding": {
4945         "description": "A string specifying the encoding format of the data contained in
4946 the optdata.",
4947         "x-detail-desc": [
4948           "oic.sec.encoding.pem - Encoding for PEM encoded certificate or chain."
4949         ],
4950         "enum": [
4951           "oic.sec.encoding.pem"
4952         ],
4953         "type": "string"
4954       },
4955       "revstat": {
4956         "description": "Revocation status flag - true = revoked.",
4957         "type": "boolean"
4958       }
4959     },
4960     "required": [
4961       "revstat"
4962     ],
4963     "type": "object"
4964   },
4965   "period": {
4966     "description": "String with RFC5545 Period.",
4967     "type": "string"
4968   },
4969   "privatedata": {
4970     "description": "Private credential information\nCredential Resource non-public
4971 contents.",
4972     "properties": {
4973       "data": {
4974         "description": "The encoded value.",
4975         "maxLength": 3072,
4976         "type": "string"
4977       },
4978       "encoding": {
4979         "description": "A string specifying the encoding format of the data contained in
4980 the privdata.",
4981         "x-detail-desc": [
4982           "oic.sec.encoding.pem - Encoding for PEM encoded private key.",
4983           "oic.sec.encoding.base64 - Encoding for Base64 encoded PSK.",

```

```

4984         "oic.sec.encoding.handle - Data is contained in a storage sub-system
4985 referenced using a handle.",
4986         "oic.sec.encoding.raw - Raw hex encoded data."
4987     ],
4988     "enum": [
4989         "oic.sec.encoding.pem",
4990         "oic.sec.encoding.base64",
4991         "oic.sec.encoding.handle",
4992         "oic.sec.encoding.raw"
4993     ],
4994     "type": "string"
4995 },
4996 "handle": {
4997     "description": "Handle to a key storage Resource.",
4998     "type": "integer"
4999 }
5000 },
5001 "required": [
5002     "encoding"
5003 ],
5004 "type": "object"
5005 },
5006 "publicdata": {
5007     "description": "Public credential information.",
5008     "properties": {
5009         "data": {
5010             "description": "The encoded value.",
5011             "maxLength": 3072,
5012             "type": "string"
5013         },
5014         "encoding": {
5015             "description": "A string specifying the encoding format of the data contained in
5016 the pubdata.",
5017             "x-detail-desc": [
5018                 "oic.sec.encoding.pem - Encoding for PEM encoded certificate or chain."
5019             ],
5020             "enum": [
5021                 "oic.sec.encoding.pem"
5022             ],
5023             "type": "string"
5024         }
5025     },
5026     "type": "object"
5027 },
5028 "roleid": {
5029     "description": "The role this credential possesses\nSecurity role specified as an
5030 <Authority> & <Rolename>. A NULL <Authority> refers to the local entity or Device.",
5031     "properties": {
5032         "authority": {
5033             "description": "The Authority component of the entity being identified. A NULL
5034 <Authority> refers to the local entity or Device.",
5035             "type": "string"
5036         },
5037         "role": {
5038             "description": "The ID of the role being identified.",
5039             "type": "string"
5040         }
5041     },
5042     "required": [
5043         "role"
5044     ],
5045     "type": "object"
5046 },
5047 "subjectuuid": {
5048     "anyOf": [
5049         {
5050             "description": "The id of the Device, which the cred entry applies to or \"*\n
5051 for wildcard identity.",
5052             "pattern": "^\\*?$",
5053             "type": "string"
5054         }
5055     ],

```

```

5055         {
5056             "description": "Format pattern according to IETF RFC 4122.",
5057             "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-
5058 F0-9]{12}$",
5059             "type": "string"
5060         }
5061     ]
5062 }
5063 },
5064 "type": "object"
5065 },
5066 "type": "array"
5067 },
5068 "if" : {
5069     "description": "The interface set supported by this Resource.",
5070     "items": {
5071         "enum": [
5072             "oic.if.baseline"
5073         ],
5074         "type": "string"
5075     },
5076     "minItems": 1,
5077     "readOnly": true,
5078     "type": "array"
5079 }
5080 },
5081 "type" : "object",
5082 "required": ["creds", "rowneruuid"]
5083 },
5084 "Cred-Update" : {
5085     "properties": {
5086         "rowneruuid" : {
5087             "description": "Format pattern according to IETF RFC 4122.",
5088             "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-
5089 9]{12}$",
5090             "type": "string"
5091         },
5092         "creds" : {
5093             "description": "List of credentials available at this Resource.",
5094             "items": {
5095                 "properties": {
5096                     "credid": {
5097                         "description": "Local reference to a credential Resource.",
5098                         "type": "integer"
5099                     },
5100                     "credtype": {
5101                         "description": "Representation of this credential's type\nCredential Types - Cred
5102 type encoded as a bitmask.0 - Empty credential used for testing1 - Symmetric pair-wise key2 -
5103 Symmetric group key4 - Asymmetric signing key8 - Asymmetric signing key with certificate16 - PIN or
5104 password32 - Asymmetric encryption key.",
5105                         "maximum": 63,
5106                         "minimum": 0,
5107                         "type": "integer"
5108                     },
5109                     "credusage": {
5110                         "description": "A string that provides hints about how/where the cred is used\nThe
5111 type of credusage.oic.sec.cred.trustca - Trust certificateoic.sec.cred.cert -
5112 Certificateoic.sec.cred.rolecert - Role Certificateoic.sec.cred.mfgtrustca - Manufacturer
5113 Certificate Trust Anchoroic.sec.cred.mfgcert - Manufacturer Certificate.",
5114                         "enum": [
5115                             "oic.sec.cred.trustca",
5116                             "oic.sec.cred.cert",
5117                             "oic.sec.cred.rolecert",
5118                             "oic.sec.cred.mfgtrustca",
5119                             "oic.sec.cred.mfgcert"
5120                         ],
5121                         "type": "string"
5122                     },
5123                     "crms": {
5124                         "description": "The refresh methods that may be used to update this credential.",
5125                         "items": {

```

```

5126         "description": "Each enum represents a method by which the credentials are
5127 refreshed.oic.sec.crm.pro - Credentials refreshed by a provisioning serviceoic.sec.crm.rdp -
5128 Credentials refreshed by a key agreement protocol and random PINoic.sec.crm.psk - Credentials
5129 refreshed by a key agreement protocoloic.sec.crm.skdc - Credentials refreshed by a key distribution
5130 serviceoic.sec.crm.pk10 - Credentials refreshed by a PKCS#10 request to a CA.",
5131         "enum": [
5132             "oic.sec.crm.pro",
5133             "oic.sec.crm.psk",
5134             "oic.sec.crm.rdp",
5135             "oic.sec.crm.skdc",
5136             "oic.sec.crm.pk10"
5137         ],
5138         "type": "string"
5139     },
5140     "type": "array"
5141 },
5142 "optionaldata": {
5143     "description": "Credential revocation status information\nOptional credential
5144 contents describes revocation status for this credential.",
5145     "properties": {
5146         "data": {
5147             "description": "The encoded structure.",
5148             "type": "string"
5149         },
5150         "encoding": {
5151             "description": "A string specifying the encoding format of the data contained in
5152 the optdata.",
5153             "x-detail-desc": [
5154                 "oic.sec.encoding.pem - Encoding for PEM encoded certificate or chain."
5155             ],
5156             "enum": [
5157                 "oic.sec.encoding.pem"
5158             ],
5159             "type": "string"
5160         },
5161         "revstat": {
5162             "description": "Revocation status flag - true = revoked.",
5163             "type": "boolean"
5164         }
5165     },
5166     "required": [
5167         "revstat"
5168     ],
5169     "type": "object"
5170 },
5171 "period": {
5172     "description": "String with RFC5545 Period.",
5173     "type": "string"
5174 },
5175 "privatedata": {
5176     "description": "Private credential information\nCredential Resource non-public
5177 contents.",
5178     "properties": {
5179         "data": {
5180             "description": "The encoded value.",
5181             "maxLength": 3072,
5182             "type": "string"
5183         },
5184         "encoding": {
5185             "description": "A string specifying the encoding format of the data contained in
5186 the privdata.",
5187             "x-detail-desc": [
5188                 "oic.sec.encoding.pem - Encoding for PEM encoded private key.",
5189                 "oic.sec.encoding.base64 - Encoding for Base64 encoded PSK.",
5190                 "oic.sec.encoding.handle - Data is contained in a storage sub-system
5191 referenced using a handle.",
5192                 "oic.sec.encoding.raw - Raw hex encoded data."
5193             ],
5194             "enum": [
5195                 "oic.sec.encoding.pem",
5196                 "oic.sec.encoding.base64",

```

```

5197         "oic.sec.encoding.handle",
5198         "oic.sec.encoding.raw"
5199     ],
5200     "type": "string"
5201 },
5202 "handle": {
5203     "description": "Handle to a key storage Resource.",
5204     "type": "integer"
5205 }
5206 },
5207 "required": [
5208     "encoding"
5209 ],
5210 "type": "object"
5211 },
5212 "publicdata": {
5213     "properties": {
5214         "data": {
5215             "description": "The encoded value.",
5216             "maxLength": 3072,
5217             "type": "string"
5218         },
5219         "encoding": {
5220             "description": "Public credential information\nA string specifying the encoding
5221 format of the data contained in the pubdata.",
5222             "x-detail-desc": [
5223                 "oic.sec.encoding.pem - Encoding for PEM encoded certificate or chain."
5224             ],
5225             "enum": [
5226                 "oic.sec.encoding.pem"
5227             ],
5228             "type": "string"
5229         }
5230     },
5231     "type": "object"
5232 },
5233 "roleid": {
5234     "description": "The role this credential possesses\nSecurity role specified as an
5235 <Authority> & <Rolename>. A NULL <Authority> refers to the local entity or Device.",
5236     "properties": {
5237         "authority": {
5238             "description": "The Authority component of the entity being identified. A NULL
5239 <Authority> refers to the local entity or Device.",
5240             "type": "string"
5241         },
5242         "role": {
5243             "description": "The ID of the role being identified.",
5244             "type": "string"
5245         }
5246     },
5247     "required": [
5248         "role"
5249 ],
5250     "type": "object"
5251 },
5252 "subjectuuid": {
5253     "anyOf": [
5254         {
5255             "description": "The id of the Device, which the cred entry applies to or \"*\n
5256 for wildcard identity.",
5257             "pattern": "^\\*$",
5258             "type": "string"
5259         },
5260         {
5261             "description": "Format pattern according to IETF RFC 4122.",
5262             "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-
5263 F0-9]{12}$",
5264             "type": "string"
5265         }
5266     ]
5267 }

```

```

5268     },
5269     "type": "object"
5270   },
5271   "type": "array"
5272 },
5273 "if" :
5274 {
5275   "description": "The interface set supported by this Resource.",
5276   "items": {
5277     "enum": [
5278       "oic.if.baseline"
5279     ],
5280     "type": "string"
5281   },
5282   "minItems": 1,
5283   "readOnly": true,
5284   "type": "array"
5285 }
5286 },
5287 "type" : "object"
5288 }
5289 }
5290 }
5291

```

### 5292 C.3.5 Property definition

5293 Table C-3 defines the Properties that are part of the "oic.r.cred" Resource Type.

5294 **Table C-3 – The Property definitions of the Resource with type "rt" = "oic.r.cred".**

Property name	Value type	Mandatory	Access mode	Description
rowneruid	string	Yes	Read Write	Format pattern according to IETF RFC 4122.
rt	array: see schema	No	Read Only	Resource Type of the Resource.
n	multiple types: see schema	No	Read Write	
id	multiple types: see schema	No	Read Write	
creds	array: see schema	Yes	Read Write	List of credentials available at this Resource.
if	array: see schema	No	Read Only	The interface set supported by this Resource.
rowneruid	string	No	Read Write	Format pattern according to IETF RFC 4122.
creds	array: see schema	No	Read Write	List of credentials available at this Resource.
if	array: see schema	No	Read Only	The interface set supported by this Resource.

### 5295 C.3.6 CRUDN behaviour

5296 Table C-4 defines the CRUDN operations that are supported on the "oic.r.cred" Resource Type.

5297

**Table C-4 – The CRUDN operations of the Resource with type "rt" = "oic.r.cred".**

Create	Read	Update	Delete	Notify
	get	post	delete	observe

5298

**C.4 Certificate Signing Request**

5299

**C.4.1 Introduction**

5300

This Resource specifies a Certificate Signing Request.

5301

5302

**C.4.2 Well-known URI**

5303

/oic/sec/csr

5304

**C.4.3 Resource type**

5305

The Resource Type is defined as: "oic.r.csr".

5306

**C.4.4 OpenAPI 2.0 definition**

5307

```

5308 {
5309   "swagger": "2.0",
5310   "info": {
5311     "title": "Certificate Signing Request",
5312     "version": "v1.0-20150819",
5313     "license": {
5314       "name": "OCF Data Model License",
5315       "url":
5316         "https://github.com/openconnectivityfoundation/core/blob/e28a9e0a92e17042ba3e83661e4c0fbce8bdc4ba/LI
5317         CENSE.md",
5318       "x-copyright": "copyright 2016-2017, 2019 Open Connectivity Foundation, Inc. All rights
5319       reserved."
5320     },
5321     "termsOfService": "https://openconnectivityfoundation.github.io/core/DISCLAIMER.md"
5322   },
5323   "schemes": ["http"],
5324   "consumes": ["application/json"],
5325   "produces": ["application/json"],
5326   "paths": {
5327     "/oic/sec/csr" : {
5328       "get": {
5329         "description": "This Resource specifies a Certificate Signing Request.\n",
5330         "parameters": [
5331           {"$ref": "#/parameters/interface"}
5332         ],
5333         "responses": {
5334           "200": {
5335             "description": "",
5336             "x-example":
5337               {
5338                 "rt": ["oic.r.csr"],
5339                 "encoding": "oic.sec.encoding.pem",
5340                 "csr": "PEMENCODEDCSR"
5341               },
5342             "schema": { "$ref": "#/definitions/Csr" }
5343           },
5344           "404": {
5345             "description": "The Device does not support certificates and generating CSRs."
5346           },
5347           "503": {
5348             "description": "The Device is not yet ready to return a response. Try again later."
5349           }
5350         }
5351       }
5352     },
5353     "parameters": {

```

```

5354     "interface" : {
5355         "in" : "query",
5356         "name" : "if",
5357         "type" : "string",
5358         "enum" : ["oic.if.baseline"]
5359     }
5360 },
5361 "definitions": {
5362     "Csr" : {
5363         "properties": {
5364             "rt" : {
5365                 "description": "Resource Type of the Resource.",
5366                 "items": {
5367                     "maxLength": 64,
5368                     "type": "string",
5369                     "enum": ["oic.r.csr"]
5370                 },
5371                 "minItems": 1,
5372                 "readOnly": true,
5373                 "type": "array"
5374             },
5375             "encoding": {
5376                 "description": "A string specifying the encoding format of the data contained in CSR.",
5377                 "x-detail-desc": [
5378                     "oic.sec.encoding.pem - Encoding for PEM encoded CSR."
5379                 ],
5380                 "enum": [
5381                     "oic.sec.encoding.pem"
5382                 ],
5383                 "readOnly": true,
5384                 "type": "string"
5385             },
5386             "n": {
5387                 "$ref":
5388                 "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
5389                 schema.json#/definitions/n"
5390             },
5391             "id": {
5392                 "$ref":
5393                 "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
5394                 schema.json#/definitions/id"
5395             },
5396             "csr": {
5397                 "description": "Signed CSR in ASN.1 in the encoding specified by the encoding property.",
5398                 "maxLength": 3072,
5399                 "readOnly": true,
5400                 "type": "string"
5401             },
5402             "if": {
5403                 "description": "The interface set supported by this Resource.",
5404                 "items": {
5405                     "enum": [
5406                         "oic.if.baseline"
5407                     ],
5408                     "type": "string"
5409                 },
5410                 "minItems": 1,
5411                 "readOnly": true,
5412                 "type": "array"
5413             }
5414         },
5415         "type" : "object",
5416         "required": ["csr", "encoding"]
5417     }
5418 }
5419 }
5420

```

#### 5421 C.4.5 Property definition

5422 Table C-5 defines the Properties that are part of the "oic.r.csr" Resource Type.



**Table C-5 – The Property definitions of the Resource with type "rt" = "oic.r.csr".**

Property name	Value type	Mandatory	Access mode	Description
rt	array: see schema	No	Read Only	Resource Type of the Resource.
encoding	string	Yes	Read Only	A string specifying the encoding format of the data contained in CSR.
n	multiple types: see schema	No	Read Write	
id	multiple types: see schema	No	Read Write	
csr	string	Yes	Read Only	Signed CSR in ASN.1 in the encoding specified by the encoding property.
if	array: see schema	No	Read Only	The interface set supported by this Resource.

**5424 C.4.6 CRUDN behaviour**

5425 Table C-6 defines the CRUDN operations that are supported on the "oic.r.csr" Resource Type.

**5426 Table C-6 – The CRUDN operations of the Resource with type "rt" = "oic.r.csr".**

Create	Read	Update	Delete	Notify
	get			observe

**5427 C.5 Device Owner Transfer Method****5428 C.5.1 Introduction**

5429 This Resource specifies properties needed to establish a Device owner.

5430

**5431 C.5.2 Well-known URI**

5432 /oic/sec/doxm

**5433 C.5.3 Resource type**

5434 The Resource Type is defined as: "oic.r.doxm".

**5435 C.5.4 OpenAPI 2.0 definition**

```

5436 {
5437   "swagger": "2.0",
5438   "info": {
5439     "title": "Device Owner Transfer Method",
5440     "version": "v1.0-20181001",
5441     "license": {
5442       "name": "OCF Data Model License",
5443       "url":
5444         "https://github.com/openconnectivityfoundation/core/blob/e28a9e0a92e17042ba3e83661e4c0fbce8bdc4ba/LI
5445         CENSE.md",
5446       "x-copyright": "copyright 2016-2017, 2019 Open Connectivity Foundation, Inc. All rights

```

```

5447 reserved."
5448 },
5449 "termsOfService": "https://openconnectivityfoundation.github.io/core/DISCLAIMER.md"
5450 },
5451 "schemes": ["http"],
5452 "consumes": ["application/json"],
5453 "produces": ["application/json"],
5454 "paths": {
5455   "/oic/sec/doxm" : {
5456     "get": {
5457       "description": "This Resource specifies properties needed to establish a Device owner.\n",
5458       "parameters": [
5459         { "$ref": "#/parameters/interface" }
5460       ],
5461       "responses": {
5462         "200": {
5463           "description": "",
5464           "x-example":
5465             {
5466               "rt": ["oic.r.doxm"],
5467               "oxms": [ 0, 2, 3 ],
5468               "oxmsel": 0,
5469               "sct": 16,
5470               "owned": true,
5471               "deviceuuid": "de305d54-75b4-431b-adb2-eb6b9e546014",
5472               "devowneruuid": "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
5473               "rowneruuid": "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9"
5474             }
5475           ,
5476           "schema": { "$ref": "#/definitions/Doxm" }
5477         },
5478         "400": {
5479           "description": "The request is invalid."
5480         }
5481       }
5482     },
5483     "post": {
5484       "description": "Updates the DOXM Resource data.\n",
5485       "parameters": [
5486         { "$ref": "#/parameters/interface" },
5487         {
5488           "name": "body",
5489           "in": "body",
5490           "required": true,
5491           "schema": { "$ref": "#/definitions/Doxm-Update" },
5492           "x-example":
5493             {
5494               "oxmsel": 0,
5495               "owned": true,
5496               "deviceuuid": "de305d54-75b4-431b-adb2-eb6b9e546014",
5497               "devowneruuid": "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
5498               "rowneruuid": "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9"
5499             }
5500         }
5501       ],
5502       "responses": {
5503         "400": {
5504           "description": "The request is invalid."
5505         },
5506         "204": {
5507           "description": "The DOXM entry is updated."
5508         }
5509       }
5510     }
5511   }
5512 },
5513 "parameters": {
5514   "interface" : {
5515     "in" : "query",
5516     "name" : "if",
5517     "type" : "string",

```

```

5518         "enum" : ["oic.if.baseline"]
5519     }
5520 },
5521 "definitions": {
5522     "Doxm" : {
5523         "properties": {
5524             "rowneruuid": {
5525                 "description": "Format pattern according to IETF RFC 4122.",
5526                 "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-
5527 9]{12}$",
5528                 "type": "string"
5529             },
5530             "oxms": {
5531                 "description": "List of supported owner transfer methods.",
5532                 "items": {
5533                     "description": "The Device owner transfer methods that may be selected at Device on-
5534 boarding. Each value indicates a specific Owner Transfer method0 - Numeric OTM identifier for the
5535 Just-Works method (oic.sec.doxm.jw)1 - Numeric OTM identifier for the random PIN method
5536 (oic.sec.doxm.rdp)2 - Numeric OTM identifier for the manufacturer certificate method
5537 (oic.sec.doxm.mfgcert)3 - Numeric OTM identifier for the decap method (oic.sec.doxm.dcap)
5538 (deprecated).",
5539                     "type": "integer"
5540                 },
5541                 "readOnly": true,
5542                 "type": "array"
5543             },
5544             "devowneruuid": {
5545                 "description": "Format pattern according to IETF RFC 4122.",
5546                 "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-
5547 9]{12}$",
5548                 "type": "string"
5549             },
5550             "deviceuuid": {
5551                 "description": "The uuid formatted identity of the Device\nFormat pattern according to
5552 IETF RFC 4122.",
5553                 "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-
5554 9]{12}$",
5555                 "type": "string"
5556             },
5557             "owned": {
5558                 "description": "Ownership status flag.",
5559                 "type": "boolean"
5560             },
5561             "n": {
5562                 "$ref":
5563 "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
5564 schema.json#/definitions/n"
5565             },
5566             "id": {
5567                 "$ref":
5568 "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
5569 schema.json#/definitions/id"
5570             },
5571             "oxmsel": {
5572                 "description": "The selected owner transfer method used during on-boarding\nThe Device
5573 owner transfer methods that may be selected at Device on-boarding. Each value indicates a specific
5574 Owner Transfer method0 - Numeric OTM identifier for the Just-Works method (oic.sec.doxm.jw)1 -
5575 Numeric OTM identifier for the random PIN method (oic.sec.doxm.rdp)2 - Numeric OTM identifier for
5576 the manufacturer certificate method (oic.sec.doxm.mfgcert)3 - Numeric OTM identifier for the decap
5577 method (oic.sec.doxm.dcap) (deprecated).",
5578                 "type": "integer"
5579             },
5580             "sct": {
5581                 "description": "Bitmask encoding of supported credential types\nCredential Types -
5582 Cred type encoded as a bitmask.0 - Empty credential used for testing1 - Symmetric pair-wise key2 -
5583 Symmetric group key4 - Asymmetric signing key8 - Asymmetric signing key with certificatel6 - PIN or
5584 password32 - Asymmetric encryption key.",
5585                 "maximum": 63,
5586                 "minimum": 0,
5587                 "type": "integer",
5588                 "readOnly": true

```

```

5589     },
5590     "rt" : {
5591         "description": "Resource Type of the Resource.",
5592         "items": {
5593             "maxLength": 64,
5594             "type": "string",
5595             "enum": ["oic.r.doxm"]
5596         },
5597         "minItems": 1,
5598         "readOnly": true,
5599         "type": "array"
5600     },
5601     "if": {
5602         "description": "The interface set supported by this Resource.",
5603         "items": {
5604             "enum": [
5605                 "oic.if.baseline"
5606             ],
5607             "type": "string"
5608         },
5609         "minItems": 1,
5610         "readOnly": true,
5611         "type": "array"
5612     }
5613 },
5614 "type" : "object",
5615 "required": ["oxms", "oxmsel", "sct", "owned", "deviceuuid", "devowneruuid", "rowneruuid"]
5616 },
5617 "Doxm-Update" : {
5618     "properties": {
5619         "rowneruuid": {
5620             "description": "Format pattern according to IETF RFC 4122.",
5621             "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{12}$",
5622             "type": "string"
5623         },
5624         "devowneruuid": {
5625             "description": "Format pattern according to IETF RFC 4122.",
5626             "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{12}$",
5627             "type": "string"
5628         },
5629         "deviceuuid": {
5630             "description": "The uuid formatted identity of the Device\nFormat pattern according to
5631 IETF RFC 4122.",
5632             "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{12}$",
5633             "type": "string"
5634         },
5635         "owned": {
5636             "description": "Ownership status flag.",
5637             "type": "boolean"
5638         },
5639         "oxmsel": {
5640             "description": "The selected owner transfer method used during on-boarding\nThe Device
5641 owner transfer methods that may be selected at Device on-boarding. Each value indicates a specific
5642 Owner Transfer method0 - Numeric OTM identifier for the Just-Works method (oic.sec.doxm.jw)1 -
5643 Numeric OTM identifier for the random PIN method (oic.sec.doxm.rdp)2 - Numeric OTM identifier for
5644 the manufacturer certificate method (oic.sec.doxm.mfgcert)3 - Numeric OTM identifier for the decap
5645 method (oic.sec.doxm.dcap) (deprecated).",
5646             "type": "integer"
5647         }
5648     }
5649 },
5650 "type" : "object"
5651 }
5652 }
5653 }
5654 }
5655 }
5656 }

```

5657 **C.5.5 Property definition**

5658 Table C-7 defines the Properties that are part of the "oic.r.doxm" Resource Type.

5659 **Table C-7 – The Property definitions of the Resource with type "rt" = "oic.r.doxm".**

Property name	Value type	Mandatory	Access mode	Description
rowneruuid	string	Yes	Read Write	Format pattern according to IETF RFC 4122.
oxms	array: see schema	Yes	Read Only	List of supported owner transfer methods.
devowneruuid	string	Yes	Read Write	Format pattern according to IETF RFC 4122.
deviceuuid	string	Yes	Read Write	The uuid formatted identity of the Device Format pattern according to IETF RFC 4122.
owned	boolean	Yes	Read Write	Ownership status flag.
n	multiple types: see schema	No	Read Write	
id	multiple types: see schema	No	Read Write	
oxmsel	integer	Yes	Read Write	The selected owner transfer method used during on-boarding The Device owner transfer methods that may be selected at Device on-boarding. Each value indicates a specific Owner Transfer method0 - Numeric OTM identifier for the Just-Works method (oic.sec.doxm.jw)1 - Numeric OTM identifier for the random PIN method (oic.sec.doxm.rdp)2 - Numeric OTM identifier for the manufacturer certificate method (oic.sec.doxm.mfgcert)3 - Numeric OTM identifier for the decap method (oic.sec.doxm.dcap) (deprecated).
sct	integer	Yes	Read Only	Bitmask encoding of supported credential types Credential Types - Cred type encoded as a

				bitmask.0 - Empty credential used for testing 1 - Symmetric pair-wise key 2 - Symmetric group key 4 - Asymmetric signing key 8 - Asymmetric signing key with certificate 16 - PIN or password 32 - Asymmetric encryption key.
rt	array: see schema	No	Read Only	Resource Type of the Resource.
if	array: see schema	No	Read Only	The interface set supported by this Resource.
rowneruuid	string		Read Write	Format pattern according to IETF RFC 4122.
devowneruuid	string		Read Write	Format pattern according to IETF RFC 4122.
deviceuuid	string		Read Write	The uuid formatted identity of the Device Format pattern according to IETF RFC 4122.
owned	boolean		Read Write	Ownership status flag.
oxmsel	integer		Read Write	The selected owner transfer method used during on-boarding The Device owner transfer methods that may be selected at Device on-boarding. Each value indicates a specific Owner Transfer method 0 - Numeric OTM identifier for the Just-Works method (oic.sec.doxm.jw) 1 - Numeric OTM identifier for the random PIN method (oic.sec.doxm.rdp) 2 - Numeric OTM identifier for the manufacturer certificate method (oic.sec.doxm.mfgcert) 3 - Numeric OTM identifier for the decap method (oic.sec.doxm.dcap) (deprecated).

5660 **C.5.6 CRUDN behaviour**

5661 Table C-8 defines the CRUDN operations that are supported on the "oic.r.doxm" Resource Type.

5662 **Table C-8 – The CRUDN operations of the Resource with type "rt" = "oic.r.doxm".**

Create	Read	Update	Delete	Notify
	get	post		observe

5663 **C.6 Device Provisioning Status**

5664 **C.6.1 Introduction**

5665 This Resource specifies Device provisioning status.

5666

5667 **C.6.2 Well-known URI**

5668 /oic/sec/pstat

5669 **C.6.3 Resource type**

5670 The Resource Type is defined as: "oic.r.pstat".

5671 **C.6.4 OpenAPI 2.0 definition**

```

5672 {
5673   "swagger": "2.0",
5674   "info": {
5675     "title": "Device Provisioning Status",
5676     "version": "v1.0-20191001",
5677     "license": {
5678       "name": "OCF Data Model License",
5679       "url":
5680 "https://github.com/openconnectivityfoundation/core/blob/e28a9e0a92e17042ba3e83661e4c0fbce8bdc4ba/LI
5681 CENSE.md",
5682       "x-copyright": "copyright 2016-2017, 2019 Open Connectivity Foundation, Inc. All rights
5683 reserved."
5684     },
5685     "termsOfService": "https://openconnectivityfoundation.github.io/core/DISCLAIMER.md"
5686   },
5687   "schemes": ["http"],
5688   "consumes": ["application/json"],
5689   "produces": ["application/json"],
5690   "paths": {
5691     "/oic/sec/pstat" : {
5692       "get": {
5693         "description": "This Resource specifies Device provisioning status.\n",
5694         "parameters": [
5695           {"$ref": "#/parameters/interface"}
5696         ],
5697         "responses": {
5698           "200": {
5699             "description": "",
5700             "x-example":
5701               {
5702                 "rt": ["oic.r.pstat"],
5703                 "dos": {"s": 3, "p": true},
5704                 "isop": true,
5705                 "cm": 8,
5706                 "tm": 60,
5707                 "om": 2,
5708                 "sm": 7,
5709                 "rowneruuid": "de305d54-75b4-431b-adb2-eb6b9e546014"
5710               },
5711             "schema": { "$ref": "#/definitions/Pstat" }
5712           },
5713           "400": {

```

```

5714         "description" : "The request is invalid."
5715     }
5716 }
5717 },
5718 "post": {
5719     "description": "Sets or updates Device provisioning status data.\n",
5720     "parameters": [
5721         {"$ref": "#/parameters/interface"},
5722         {
5723             "name": "body",
5724             "in": "body",
5725             "required": true,
5726             "schema": { "$ref": "#/definitions/Pstat-Update" },
5727             "x-example":
5728                 {
5729                     "dos": {"s": 3},
5730                     "tm": 60,
5731                     "om": 2,
5732                     "rowneruuid": "de305d54-75b4-431b-adb2-eb6b9e546014"
5733                 }
5734         }
5735     ],
5736     "responses": {
5737         "400": {
5738             "description": "The request is invalid."
5739         },
5740         "204": {
5741             "description": "The PSTAT entry is updated."
5742         }
5743     }
5744 }
5745 },
5746 "parameters": {
5747     "interface": {
5748         "in": "query",
5749         "name": "if",
5750         "type": "string",
5751         "enum": ["oic.if.baseline"]
5752     }
5753 },
5754 "definitions": {
5755     "Pstat": {
5756         "properties": {
5757             "rowneruuid": {
5758                 "description": "The UUID formatted identity of the Resource owner\nFormat pattern
5759 according to IETF RFC 4122.",
5760                 "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-
5761 9]{12}$",
5762                 "type": "string"
5763             },
5764             "rt": {
5765                 "description": "Resource Type of the Resource.",
5766                 "items": {
5767                     "maxLength": 64,
5768                     "type": "string",
5769                     "enum": ["oic.r.pstat"]
5770                 },
5771                 "minItems": 1,
5772                 "readOnly": true,
5773                 "type": "array"
5774             },
5775             "om": {
5776                 "description": "Current operational mode\nDevice provisioning operation may be server
5777 directed or client (aka provisioning service) directed. The value is a bitmask encoded as integer
5778 and indicates the provisioning operation modes1 - Server-directed utilizing multiple provisioning
5779 services2 - Server-directed utilizing a single provisioning service4 - Client-directed provisioning8
5780 - Unused16 - Unused32 - Unused64 - Unused128 - Unused.",
5781                 "maximum": 7,
5782                 "minimum": 1,
5783                 "type": "integer"
5784             }
5785         }
5786     }
5787 }

```



```

5785     },
5786     "cm": {
5787       "description": "Current Device provisioning mode\nDevice provisioning mode maintains a
5788 bitmask of the possible provisioning states of a Device. The value can be either 8 or 16 character
5789 in length. If its only 8 characters it represents the lower byte value1 - Manufacturer reset state2
5790 - Device pairing and owner transfer state4 - Unused8 - Provisioning of credential management
5791 services16 - Provisioning of access management services32 - Provisioning of local ACLs64 - Initiate
5792 Software Version Validation128 - Initiate Secure Software Update.",
5793       "maximum": 255,
5794       "minimum": 0,
5795       "type": "integer",
5796       "readOnly": true
5797     },
5798     "n": {
5799       "$ref":
5800 "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
5801 schema.json#/definitions/n"
5802     },
5803     "id": {
5804       "$ref":
5805 "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
5806 schema.json#/definitions/id"
5807     },
5808     "isop": {
5809       "description": "true indicates Device is operational.",
5810       "readOnly": true,
5811       "type": "boolean"
5812     },
5813     "tm": {
5814       "description": "Target Device provisioning mode\nDevice provisioning mode maintains a
5815 bitmask of the possible provisioning states of a Device. The value can be either 8 or 16 character
5816 in length. If its only 8 characters it represents the lower byte value1 - Manufacturer reset state2
5817 - Device pairing and owner transfer state4 - Unused8 - Provisioning of credential management
5818 services16 - Provisioning of access management services32 - Provisioning of local ACLs64 - Initiate
5819 Software Version Validation128 - Initiate Secure Software Update.",
5820       "maximum": 255,
5821       "minimum": 0,
5822       "type": "integer"
5823     },
5824     "sm": {
5825       "description": "Supported operational modes\nDevice provisioning operation may be server
5826 directed or client (aka provisioning service) directed. The value is a bitmask encoded as integer
5827 and indicates the provisioning operation modes1 - Server-directed utilizing multiple provisioning
5828 services2 - Server-directed utilizing a single provisioning service4 - Client-directed provisioning8
5829 - Unused16 - Unused32 - Unused64 - Unused128 - Unused.",
5830       "maximum": 7,
5831       "minimum": 1,
5832       "type": "integer",
5833       "readOnly": true
5834     },
5835     "dos": {
5836       "description": "Device on-boarding state\nDevice operation state machine.",
5837       "properties": {
5838         "p": {
5839           "default": true,
5840           "description": "'p' is TRUE when the 's' state is pending until all necessary changes
5841 to Device Resources are complete.",
5842           "readOnly": true,
5843           "type": "boolean"
5844         },
5845         "s": {
5846           "description": "The current or pending operational state.",
5847           "x-detail-desc": [
5848             "0 - RESET - Device reset state.",
5849             "1 - RFOFM - Ready for Device owner transfer method state.",
5850             "2 - RFPRO - Ready for Device provisioning state.",
5851             "3 - RFNOP - Ready for Device normal operation state.",
5852             "4 - SRESET - The Device is in a soft reset state."
5853           ],
5854           "maximum": 4,
5855           "minimum": 0,

```

```

5856         "type": "integer"
5857     }
5858 },
5859 "required": [
5860     "s"
5861 ],
5862 "type": "object"
5863 },
5864 "if" : {
5865     "description": "The interface set supported by this Resource.",
5866     "items": {
5867         "enum": [
5868             "oic.if.baseline"
5869         ],
5870         "type": "string"
5871     },
5872     "minItems": 1,
5873     "readOnly": true,
5874     "type": "array"
5875 }
5876 },
5877 "type" : "object",
5878 "required": ["dos", "isop", "cm", "tm", "om", "sm", "rowneruuid"]
5879 },
5880 "Pstat-Update" : {
5881     "properties": {
5882         "rowneruuid": {
5883             "description": "The UUID formatted identity of the Resource owner\nFormat pattern
5884 according to IETF RFC 4122.",
5885             "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-
5886 9]{12}$",
5887             "type": "string"
5888         },
5889         "om": {
5890             "description": "Current operational mode\nDevice provisioning operation may be server
5891 directed or client (aka provisioning service) directed. The value is a bitmask encoded as integer
5892 and indicates the provisioning operation modes1 - Server-directed utilizing multiple provisioning
5893 services2 - Server-directed utilizing a single provisioning service4 - Client-directed provisioning8
5894 - Unused16 - Unused32 - Unused64 - Unused128 - Unused.",
5895             "maximum": 7,
5896             "minimum": 1,
5897             "type": "integer"
5898         },
5899         "tm": {
5900             "description": "Target Device provisioning mode\nDevice provisioning mode maintains a
5901 bitmask of the possible provisioning states of a Device. The value can be either 8 or 16 character
5902 in length. If its only 8 characters it represents the lower byte value1 - Manufacturer reset state2
5903 - Device pairing and owner transfer state4 - Unused8 - Provisioning of credential management
5904 services16 - Provisioning of access management services32 - Provisioning of local ACLs64 - Initiate
5905 Software Version Validation128 - Initiate Secure Software Update.",
5906             "maximum": 255,
5907             "minimum": 0,
5908             "type": "integer"
5909         },
5910         "dos": {
5911             "description": "Device on-boarding state\nDevice operation state machine.",
5912             "properties": {
5913                 "p": {
5914                     "default": true,
5915                     "description": "'p' is TRUE when the 's' state is pending until all necessary changes
5916 to Device Resources are complete.",
5917                     "readOnly": true,
5918                     "type": "boolean"
5919                 },
5920                 "s": {
5921                     "description": "The current or pending operational state.",
5922                     "x-detail-desc": [
5923                         "0 - RESET - Device reset state.",
5924                         "1 - RFOFM - Ready for Device owner transfer method state.",
5925                         "2 - RFPPO - Ready for Device provisioning state.",
5926                         "3 - RFNOP - Ready for Device normal operation state.",

```

```

5927         "4 - SRESET - The Device is in a soft reset state."
5928     ],
5929     "maximum": 4,
5930     "minimum": 0,
5931     "type": "integer"
5932 }
5933 },
5934 "required": [
5935     "s"
5936 ],
5937 "type": "object"
5938 }
5939 },
5940 "type" : "object"
5941 }
5942 }
5943 }
5944

```

5945 **C.6.5 Property definition**

5946 Table C-9 defines the Properties that are part of the "oic.r.pstat" Resource Type.

5947 **Table C-9 – The Property definitions of the Resource with type "rt" = "oic.r.pstat".**

Property name	Value type	Mandatory	Access mode	Description
rowneruuid	string	Yes	Read Write	The UUID formatted identity of the Resource owner. Format pattern according to IETF RFC 4122.
rt	array: see schema	No	Read Only	Resource Type of the Resource.
om	integer	Yes	Read Write	Current operational mode. Device provisioning operation may be server directed or client (aka provisioning service) directed. The value is a bitmask encoded as integer and indicates the provisioning operation modes: modes1 - Server-directed utilizing multiple provisioning services services2 - Server-directed utilizing a single provisioning service service4 - Client-directed

				provisioning8 - Unused16 - Unused32 - Unused64 - Unused128 - Unused.
cm	integer	Yes	Read Only	Current Device provisioning mode Device provisioning mode maintains a bitmask of the possible provisioning states of a Device. The value can be either 8 or 16 character in length. If its only 8 characters it represents the lower byte value1 - Manufacturer reset state2 - Device pairing and owner transfer state4 - Unused8 - Provisioning of credential management services16 - Provisioning of access management services32 - Provisioning of local ACLs64 - Initiate Software Version Validation128 - Initiate Secure Software Update.
n	multiple types: see schema	No	Read Write	
id	multiple types: see schema	No	Read Write	
isop	boolean	Yes	Read Only	true indicates Device is operational.
tm	integer	Yes	Read Write	Target Device provisioning

				<p>mode Device provisioning mode maintains a bitmask of the possible provisioning states of a Device. The value can be either 8 or 16 character in length. If its only 8 characters it represents the lower byte value1 - Manufacturer reset state2 - Device pairing and owner transfer state4 - Unused8 - Provisioning of credential management services16 - Provisioning of access management services32 - Provisioning of local ACLs64 - Initiate Software Version Validation128 - Initiate Secure Software Update.</p>
sm	integer	Yes	Read Only	<p>Supported operational modes Device provisioning operation may be server directed or client (aka provisioning service) directed. The value is a bitmask encoded as integer and indicates the provisioning operation modes1 - Server-</p>

				directed utilizing multiple provisioning services2 - Server-directed utilizing a single provisioning service4 - Client-directed provisioning8 - Unused16 - Unused32 - Unused64 - Unused128 - Unused.
dos	object: schema see	Yes	Read Write	Device onboarding state Device operation state machine.
if	array: schema see	No	Read Only	The interface set supported by this Resource.
owneruuid	string	No	Read Write	The UUID formatted identity of the Resource owner Format pattern according to IETF RFC 4122.
om	integer	No	Read Write	Current operational mode Device provisioning operation may be server directed or client (aka provisioning service) directed. The value is a bitmask encoded as integer and indicates the provisioning operation modes1 - Server-directed utilizing multiple provisioning services2 - Server-directed utilizing a single provisioning service4 - Client-directed provisioning8 -

				Unused16 - Unused32 - Unused64 - Unused128 - Unused.
tm	integer	No	Read Write	Target Device provisioning mode Device provisioning mode maintains a bitmask of the possible provisioning states of a Device. The value can be either 8 or 16 character in length. If its only 8 characters it represents the lower byte value1 - Manufacturer reset state2 - Device pairing and owner transfer state4 - Unused8 - Provisioning of credential management services16 - Provisioning of access management services32 - Provisioning of local ACLs64 - Initiate Software Version Validation128 - Initiate Secure Software Update.
dos	object: see schema	No	Read Write	Device on-boarding state Device operation state machine.

5948 **C.6.6 CRUDN behaviour**

5949 Table C-10 defines the CRUDN operations that are supported on the "oic.r.pstat" Resource Type.

5950 **Table C-10 – The CRUDN operations of the Resource with type "rt" = "oic.r.pstat".**

Create	Read	Update	Delete	Notify
	get	post		observe

5951 **C.7 Asserted Roles**

5952 **C.7.1 Introduction**

5953 This Resource specifies roles that have been asserted.  
5954

5955 **C.7.2 Well-known URI**

5956 /oic/sec/roles

5957 **C.7.3 Resource type**

5958 The Resource Type is defined as: "oic.r.roles".

5959 **C.7.4 OpenAPI 2.0 definition**

```

5960 {
5961   "swagger": "2.0",
5962   "info": {
5963     "title": "Asserted Roles",
5964     "version": "v1.0-20170323",
5965     "license": {
5966       "name": "OCF Data Model License",
5967       "url":
5968         "https://github.com/openconnectivityfoundation/core/blob/e28a9e0a92e17042ba3e83661e4c0fbce8bdc4ba/LI
5969         CENSE.md",
5970       "x-copyright": "copyright 2016-2017, 2019 Open Connectivity Foundation, Inc. All rights
5971         reserved."
5972     },
5973     "termsOfService": "https://openconnectivityfoundation.github.io/core/DISCLAIMER.md"
5974   },
5975   "schemes": ["http"],
5976   "consumes": ["application/json"],
5977   "produces": ["application/json"],
5978   "paths": {
5979     "/oic/sec/roles" : {
5980       "get": {
5981         "description": "This Resource specifies roles that have been asserted.\n",
5982         "parameters": [
5983           {"$ref": "#/parameters/interface"}
5984         ],
5985         "responses": {
5986           "200": {
5987             "description": "",
5988             "x-example":
5989               {
5990                 "roles" :[
5991                   {
5992                     "credid":1,
5993                     "credtype":8,
5994                     "subjectuuid":"00000000-0000-0000-0000-000000000000",
5995                     "publicdata":
5996                       {
5997                         "encoding":"oic.sec.encoding.pem",
5998                         "data":"PEMENCODEDROLECERT"
5999                       },
6000                     "optionaldata":
6001                       {
6002                         "revstat": false,
6003                         "encoding":"oic.sec.encoding.pem",
6004                         "data":"PEMENCODEDISSUERCERT"
6005                       }
6006                   }
6007                 ]
6008               }
6009           }
6010         }
6011       }
6012     }
6013   }

```



```

6007         {
6008             "credid":2,
6009             "credtype":8,
6010             "subjectuuiid":"00000000-0000-0000-0000-000000000000",
6011             "publicdata":
6012                 {
6013                     "encoding":"oic.sec.encoding.pem",
6014                     "data":"PEMENCODEDROLECERT"
6015                 },
6016             "optionaldata":
6017                 {
6018                     "revstat": false,
6019                     "encoding":"oic.sec.encoding.pem",
6020                     "data":"PEMENCODEDISSUERCERT"
6021                 }
6022         },
6023     ],
6024     "rt":["oic.r.roles"],
6025     "if":["oic.if.baseline"]
6026 }
6027 ,
6028 "schema": { "$ref": "#/definitions/Roles" }
6029 },
6030 "400": {
6031     "description": "The request is invalid."
6032 }
6033 },
6034 },
6035 "post": {
6036     "description": "Update the roles Resource, i.e., assert new roles to this server.\n\nNew
6037 role certificates that match an existing certificate (i.e., publicdata\nand optionaldata are the
6038 same) are not added to the Resource (and 204 is\nreturned).\n\nThe provided credid values are
6039 ignored, the Resource assigns its own.\n",
6040     "parameters": [
6041         { "$ref": "#/parameters/interface" },
6042         {
6043             "name": "body",
6044             "in": "body",
6045             "required": true,
6046             "schema": { "$ref": "#/definitions/Roles-update" },
6047             "x-example":
6048                 {
6049                     "roles" :[
6050                         {
6051                             "credid":1,
6052                             "credtype":8,
6053                             "subjectuuiid":"00000000-0000-0000-0000-000000000000",
6054                             "publicdata":
6055                                 {
6056                                     "encoding":"oic.sec.encoding.pem",
6057                                     "data":"PEMENCODEDROLECERT"
6058                                 },
6059                             "optionaldata":
6060                                 {
6061                                     "revstat": false,
6062                                     "encoding":"oic.sec.encoding.pem",
6063                                     "data":"PEMENCODEDISSUERCERT"
6064                                 }
6065                         },
6066                         {
6067                             "credid":2,
6068                             "credtype":8,
6069                             "subjectuuiid":"00000000-0000-0000-0000-000000000000",
6070                             "publicdata":
6071                                 {
6072                                     "encoding":"oic.sec.encoding.pem",
6073                                     "data":"PEMENCODEDROLECERT"
6074                                 },
6075                             "optionaldata":
6076                                 {
6077                                     "revstat": false,

```

```

6078         "encoding": "oic.sec.encoding.pem",
6079         "data": "PEMENCODEDISSUERCERT"
6080     }
6081 }
6082 ]
6083 }
6084 }
6085 ],
6086 "responses": {
6087     "400": {
6088         "description": "The request is invalid."
6089     },
6090     "204": {
6091         "description": "The roles entry is updated."
6092     }
6093 }
6094 },
6095 "delete": {
6096     "description": "Deletes roles Resource entries.\nWhen DELETE is used without query
6097 parameters, all the roles entries are deleted.\nWhen DELETE is used with a query parameter, only the
6098 entries matching\nthe query parameter are deleted.\n",
6099     "parameters": [
6100         {"$ref": "#/parameters/interface"},
6101         {"$ref": "#/parameters/roles-filtered"}
6102     ],
6103     "responses": {
6104         "200": {
6105             "description": "The specified or all roles Resource entries have been successfully
6106 deleted."
6107         },
6108         "400": {
6109             "description": "The request is invalid."
6110         }
6111     }
6112 }
6113 }
6114 },
6115 "parameters": {
6116     "interface": {
6117         "in": "query",
6118         "name": "if",
6119         "type": "string",
6120         "enum": ["oic.if.baseline"]
6121     },
6122     "roles-filtered": {
6123         "in": "query",
6124         "name": "credid",
6125         "required": false,
6126         "type": "integer",
6127         "description": "Only applies to the credential with the specified credid.",
6128         "x-example": 2112
6129     }
6130 },
6131 "definitions": {
6132     "Roles": {
6133         "properties": {
6134             "rt": {
6135                 "description": "Resource Type of the Resource.",
6136                 "items": {
6137                     "maxLength": 64,
6138                     "type": "string",
6139                     "enum": ["oic.r.roles"]
6140                 },
6141                 "minItems": 1,
6142                 "readOnly": true,
6143                 "type": "array"
6144             },
6145             "n": {
6146                 "$ref":
6147 "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
6148 schema.json#/definitions/n"

```

```

6149     },
6150     "id": {
6151       "$ref":
6152       "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
6153       schema.json#/definitions/id"
6154     },
6155     "roles": {
6156       "description": "List of role certificates.",
6157       "items": {
6158         "properties": {
6159           "credid": {
6160             "description": "Local reference to a credential Resource.",
6161             "type": "integer"
6162           },
6163           "credtype": {
6164             "description": "Representation of this credential's type\nCredential Types - Cred
6165 type encoded as a bitmask.0 - Empty credential used for testing1 - Symmetric pair-wise key2 -
6166 Symmetric group key4 - Asymmetric signing key8 - Asymmetric signing key with certificatel6 - PIN or
6167 password32 - Asymmetric encryption key.",
6168             "maximum": 63,
6169             "minimum": 0,
6170             "type": "integer"
6171           },
6172           "credusage": {
6173             "description": "A string that provides hints about how/where the cred is used\nThe
6174 type of credusage.oic.sec.cred.trustca - Trust certificateoic.sec.cred.cert -
6175 Certificateoic.sec.cred.rolecert - Role Certificateoic.sec.cred.mfgtrustca - Manufacturer
6176 Certificate Trust Anchoroic.sec.cred.mfgcert - Manufacturer Certificate.",
6177             "enum": [
6178               "oic.sec.cred.trustca",
6179               "oic.sec.cred.cert",
6180               "oic.sec.cred.rolecert",
6181               "oic.sec.cred.mfgtrustca",
6182               "oic.sec.cred.mfgcert"
6183             ],
6184             "type": "string"
6185           },
6186           "crms": {
6187             "description": "The refresh methods that may be used to update this credential.",
6188             "items": {
6189               "description": "Each enum represents a method by which the credentials are
6190 refreshed.oic.sec.crm.pro - Credentials refreshed by a provisioning serviceoic.sec.crm.rdp -
6191 Credentials refreshed by a key agreement protocol and random PINoic.sec.crm.psk - Credentials
6192 refreshed by a key agreement protocoloic.sec.crm.skdc - Credentials refreshed by a key distribution
6193 serviceoic.sec.crm.pk10 - Credentials refreshed by a PKCS#10 request to a CA.",
6194             "enum": [
6195               "oic.sec.crm.pro",
6196               "oic.sec.crm.psk",
6197               "oic.sec.crm.rdp",
6198               "oic.sec.crm.skdc",
6199               "oic.sec.crm.pk10"
6200             ],
6201             "type": "string"
6202           },
6203           "type": "array"
6204         },
6205         "optionaldata": {
6206           "description": "Credential revocation status information\nOptional credential
6207 contents describes revocation status for this credential.",
6208           "properties": {
6209             "data": {
6210               "description": "This is the encoded structure.",
6211               "type": "string"
6212             },
6213             "encoding": {
6214               "description": "A string specifying the encoding format of the data contained in
6215 the optdata.",
6216               "x-detail-desc": [
6217                 "oic.sec.encoding.jwt - RFC7517 JSON web token (JWT) encoding.",
6218                 "oic.sec.encoding.cwt - RFC CBOR web token (CWT) encoding.",
6219                 "oic.sec.encoding.base64 - Base64 encoded object.",

```

```

6220         "oic.sec.encoding.pem - Encoding for PEM encoded certificate or chain.",
6221         "oic.sec.encoding.der - Encoding for DER encoded certificate.",
6222         "oic.sec.encoding.raw - Raw hex encoded data."
6223     ],
6224     "enum": [
6225         "oic.sec.encoding.jwt",
6226         "oic.sec.encoding.cwt",
6227         "oic.sec.encoding.base64",
6228         "oic.sec.encoding.pem",
6229         "oic.sec.encoding.der",
6230         "oic.sec.encoding.raw"
6231     ],
6232     "type": "string"
6233 },
6234 "revstat": {
6235     "description": "Revocation status flag - true = revoked.",
6236     "type": "boolean"
6237 },
6238 },
6239 "required": [
6240     "revstat"
6241 ],
6242 "type": "object"
6243 },
6244 "period": {
6245     "description": "String with RFC5545 Period.",
6246     "type": "string"
6247 },
6248 "privatedata": {
6249     "description": "Private credential information\nCredential Resource non-public
6250 contents.",
6251     "properties": {
6252         "data": {
6253             "description": "The encoded value.",
6254             "maxLength": 3072,
6255             "type": "string"
6256         },
6257         "encoding": {
6258             "description": "A string specifying the encoding format of the data contained in
6259 the privdata.",
6260             "x-detail-desc": [
6261                 "oic.sec.encoding.jwt - RFC7517 JSON web token (JWT) encoding.",
6262                 "oic.sec.encoding.cwt - RFC CBOR web token (CWT) encoding.",
6263                 "oic.sec.encoding.base64 - Base64 encoded object.",
6264                 "oic.sec.encoding.uri - URI reference.",
6265                 "oic.sec.encoding.handle - Data is contained in a storage sub-system
6266 referenced using a handle.",
6267                 "oic.sec.encoding.raw - Raw hex encoded data."
6268             ],
6269             "enum": [
6270                 "oic.sec.encoding.jwt",
6271                 "oic.sec.encoding.cwt",
6272                 "oic.sec.encoding.base64",
6273                 "oic.sec.encoding.uri",
6274                 "oic.sec.encoding.handle",
6275                 "oic.sec.encoding.raw"
6276             ],
6277             "type": "string"
6278         },
6279         "handle": {
6280             "description": "Handle to a key storage Resource.",
6281             "type": "integer"
6282         }
6283     },
6284     "required": [
6285         "encoding"
6286     ],
6287     "type": "object"
6288 },
6289 "publicdata": {
6290     "description": "Public credential information.",

```

```

6291     "properties": {
6292         "data": {
6293             "description": "This is the encoded value.",
6294             "maxLength": 3072,
6295             "type": "string"
6296         },
6297         "encoding": {
6298             "description": "A string specifying the encoding format of the data contained in
6299 the pubdata.",
6300             "x-detail-desc": [
6301                 "oic.sec.encoding.jwt - RFC7517 JSON web token (JWT) encoding.",
6302                 "oic.sec.encoding.cwt - RFC CBOR web token (CWT) encoding.",
6303                 "oic.sec.encoding.base64 - Base64 encoded object.",
6304                 "oic.sec.encoding.uri - URI reference.",
6305                 "oic.sec.encoding.pem - Encoding for PEM encoded certificate or chain.",
6306                 "oic.sec.encoding.der - Encoding for DER encoded certificate.",
6307                 "oic.sec.encoding.raw - Raw hex encoded data."
6308             ],
6309             "enum": [
6310                 "oic.sec.encoding.jwt",
6311                 "oic.sec.encoding.cwt",
6312                 "oic.sec.encoding.base64",
6313                 "oic.sec.encoding.uri",
6314                 "oic.sec.encoding.pem",
6315                 "oic.sec.encoding.der",
6316                 "oic.sec.encoding.raw"
6317             ],
6318             "type": "string"
6319         }
6320     },
6321     "type": "object"
6322 },
6323 "roleid": {
6324     "description": "The role this credential possesses\nSecurity role specified as an
6325 <Authority> & <Rolename>. A NULL <Authority> refers to the local entity or Device.",
6326     "properties": {
6327         "authority": {
6328             "description": "The Authority component of the entity being identified. A NULL
6329 <Authority> refers to the local entity or Device.",
6330             "type": "string"
6331         },
6332         "role": {
6333             "description": "The ID of the role being identified.",
6334             "type": "string"
6335         }
6336     },
6337     "required": [
6338         "role"
6339     ],
6340     "type": "object"
6341 },
6342 "subjectuuid": {
6343     "anyOf": [
6344         {
6345             "description": "The id of the Device, which the cred entry applies to or \"*\
6346 for wildcard identity.",
6347             "pattern": "^[\\*]*$",
6348             "type": "string"
6349         },
6350         {
6351             "description": "Format pattern according to IETF RFC 4122.",
6352             "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-
6353 F0-9]{12}$",
6354             "type": "string"
6355         }
6356     ]
6357 },
6358 },
6359 "type": "object"
6360 },
6361 "type": "array"

```

```

6362     },
6363     "if": {
6364         "description": "The interface set supported by this Resource.",
6365         "items": {
6366             "enum": [
6367                 "oic.if.baseline"
6368             ],
6369             "type": "string"
6370         },
6371         "minItems": 1,
6372         "readOnly": true,
6373         "type": "array"
6374     }
6375 },
6376 "type": "object",
6377 "required": ["roles"]
6378 },
6379 "Roles-update" : {
6380     "properties": {
6381         "roles": {
6382             "description": "List of role certificates.",
6383             "items": {
6384                 "properties": {
6385                     "credid": {
6386                         "description": "Local reference to a credential Resource.",
6387                         "type": "integer"
6388                     },
6389                     "credtype": {
6390                         "description": "Representation of this credential's type\nCredential Types - Cred
6391 type encoded as a bitmask.0 - Empty credential used for testing1 - Symmetric pair-wise key2 -
6392 Symmetric group key4 - Asymmetric signing key8 - Asymmetric signing key with certificatel6 - PIN or
6393 password32 - Asymmetric encryption key.",
6394                         "maximum": 63,
6395                         "minimum": 0,
6396                         "type": "integer"
6397                     },
6398                     "credusage": {
6399                         "description": "A string that provides hints about how/where the cred is used\nThe
6400 type of credusage.oic.sec.cred.trustca - Trust certificateoic.sec.cred.cert -
6401 Certificateoic.sec.cred.rolecert - Role Certificateoic.sec.cred.mfgtrustca - Manufacturer
6402 Certificate Trust Anchoroic.sec.cred.mfgcert - Manufacturer Certificate.",
6403                         "enum": [
6404                             "oic.sec.cred.trustca",
6405                             "oic.sec.cred.cert",
6406                             "oic.sec.cred.rolecert",
6407                             "oic.sec.cred.mfgtrustca",
6408                             "oic.sec.cred.mfgcert"
6409                         ],
6410                         "type": "string"
6411                     },
6412                     "crms": {
6413                         "description": "The refresh methods that may be used to update this credential.",
6414                         "items": {
6415                             "description": "Each enum represents a method by which the credentials are
6416 refreshed.oic.sec.crm.pro - Credentials refreshed by a provisioning serviceoic.sec.crm.rdp -
6417 Credentials refreshed by a key agreement protocol and random PINoic.sec.crm.psk - Credentials
6418 refreshed by a key agreement protocoloic.sec.crm.skdc - Credentials refreshed by a key distribution
6419 serviceoic.sec.crm.pk10 - Credentials refreshed by a PKCS#10 request to a CA.",
6420                             "enum": [
6421                                 "oic.sec.crm.pro",
6422                                 "oic.sec.crm.psk",
6423                                 "oic.sec.crm.rdp",
6424                                 "oic.sec.crm.skdc",
6425                                 "oic.sec.crm.pk10"
6426                             ],
6427                             "type": "string"
6428                         },
6429                         "type": "array"
6430                     },
6431                     "optionaldata": {
6432                         "description": "Credential revocation status information\nOptional credential

```

```

6433 contents describes revocation status for this credential.",
6434     "properties": {
6435         "data": {
6436             "description": "This is the encoded structure.",
6437             "type": "string"
6438         },
6439         "encoding": {
6440             "description": "A string specifying the encoding format of the data contained in
6441 the optdata.",
6442             "x-detail-desc": [
6443                 "oic.sec.encoding.jwt - RFC7517 JSON web token (JWT) encoding.",
6444                 "oic.sec.encoding.cwt - RFC CBOR web token (CWT) encoding.",
6445                 "oic.sec.encoding.base64 - Base64 encoded object.",
6446                 "oic.sec.encoding.pem - Encoding for PEM encoded certificate or chain.",
6447                 "oic.sec.encoding.der - Encoding for DER encoded certificate.",
6448                 "oic.sec.encoding.raw - Raw hex encoded data."
6449             ],
6450             "enum": [
6451                 "oic.sec.encoding.jwt",
6452                 "oic.sec.encoding.cwt",
6453                 "oic.sec.encoding.base64",
6454                 "oic.sec.encoding.pem",
6455                 "oic.sec.encoding.der",
6456                 "oic.sec.encoding.raw"
6457             ],
6458             "type": "string"
6459         },
6460         "revstat": {
6461             "description": "Revocation status flag - true = revoked.",
6462             "type": "boolean"
6463         }
6464     },
6465     "required": [
6466         "revstat"
6467     ],
6468     "type": "object"
6469 },
6470 "period": {
6471     "description": "String with RFC5545 Period.",
6472     "type": "string"
6473 },
6474 "privatedata": {
6475     "description": "Private credential information\nCredential Resource non-public
6476 contents.",
6477     "properties": {
6478         "data": {
6479             "description": "The encoded value.",
6480             "maxLength": 3072,
6481             "type": "string"
6482         },
6483         "encoding": {
6484             "description": "A string specifying the encoding format of the data contained in
6485 the privdata.",
6486             "x-detail-desc": [
6487                 "oic.sec.encoding.jwt - RFC7517 JSON web token (JWT) encoding.",
6488                 "oic.sec.encoding.cwt - RFC CBOR web token (CWT) encoding.",
6489                 "oic.sec.encoding.base64 - Base64 encoded object.",
6490                 "oic.sec.encoding.uri - URI reference.",
6491                 "oic.sec.encoding.handle - Data is contained in a storage sub-system
6492 referenced using a handle.",
6493                 "oic.sec.encoding.raw - Raw hex encoded data."
6494             ],
6495             "enum": [
6496                 "oic.sec.encoding.jwt",
6497                 "oic.sec.encoding.cwt",
6498                 "oic.sec.encoding.base64",
6499                 "oic.sec.encoding.uri",
6500                 "oic.sec.encoding.handle",
6501                 "oic.sec.encoding.raw"
6502             ],
6503             "type": "string"

```

```

6504         },
6505         "handle": {
6506             "description": "Handle to a key storage Resource.",
6507             "type": "integer"
6508         }
6509     },
6510     "required": [
6511         "encoding"
6512     ],
6513     "type": "object"
6514 },
6515 "publicdata": {
6516     "description": "Public credential information.",
6517     "properties": {
6518         "data": {
6519             "description": "The encoded value.",
6520             "maxLength": 3072,
6521             "type": "string"
6522         },
6523         "encoding": {
6524             "description": "A string specifying the encoding format of the data contained in
6525 the pubdata.",
6526             "x-detail-desc": [
6527                 "oic.sec.encoding.jwt - RFC7517 JSON web token (JWT) encoding.",
6528                 "oic.sec.encoding.cwt - RFC CBOR web token (CWT) encoding.",
6529                 "oic.sec.encoding.base64 - Base64 encoded object.",
6530                 "oic.sec.encoding.uri - URI reference.",
6531                 "oic.sec.encoding.pem - Encoding for PEM encoded certificate or chain.",
6532                 "oic.sec.encoding.der - Encoding for DER encoded certificate.",
6533                 "oic.sec.encoding.raw - Raw hex encoded data."
6534             ],
6535             "enum": [
6536                 "oic.sec.encoding.jwt",
6537                 "oic.sec.encoding.cwt",
6538                 "oic.sec.encoding.base64",
6539                 "oic.sec.encoding.uri",
6540                 "oic.sec.encoding.pem",
6541                 "oic.sec.encoding.der",
6542                 "oic.sec.encoding.raw"
6543             ],
6544             "type": "string"
6545         }
6546     },
6547     "type": "object"
6548 },
6549 "roleid": {
6550     "description": "The role this credential possesses\nSecurity role specified as an
6551 <Authority> & <Rolename>. A NULL <Authority> refers to the local entity or Device.",
6552     "properties": {
6553         "authority": {
6554             "description": "The Authority component of the entity being identified. A NULL
6555 <Authority> refers to the local entity or Device.",
6556             "type": "string"
6557         },
6558         "role": {
6559             "description": "The ID of the role being identified.",
6560             "type": "string"
6561         }
6562     },
6563     "required": [
6564         "role"
6565     ],
6566     "type": "object"
6567 },
6568 "subjectuuid": {
6569     "anyOf": [
6570         {
6571             "description": "The id of the Device, which the cred entry applies to or \"*\n"
6572 for wildcard identity.",
6573             "pattern": "^[\\*]$",
6574             "type": "string"

```



```

6575         },
6576         {
6577             "description": "Format pattern according to IETF RFC 4122.",
6578             "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-
6579 F0-9]{12}$",
6580             "type": "string"
6581         }
6582     ]
6583 },
6584 },
6585 "type": "object"
6586 },
6587 "type": "array"
6588 }
6589 },
6590 "type": "object",
6591 "required": ["roles"]
6592 }
6593 }
6594 }
6595

```

### 6596 C.7.5 Property definition

6597 Table C-11 defines the Properties that are part of the "oic.r.roles" Resource Type.

6598 **Table C-11 – The Property definitions of the Resource with type "rt" = "oic.r.roles".**

Property name	Value type	Mandatory	Access mode	Description
rt	array: see schema	No	Read Only	Resource Type of the Resource.
n	multiple types: see schema	No	Read Write	
id	multiple types: see schema	No	Read Write	
roles	array: see schema	Yes	Read Write	List of role certificates.
if	array: see schema	No	Read Only	The interface set supported by this Resource.
roles	array: see schema	Yes	Read Write	List of role certificates.

### 6599 C.7.6 CRUDN behaviour

6600 Table C-12 defines the CRUDN operations that are supported on the "oic.r.roles" Resource Type.

6601 **Table C-12 – The CRUDN operations of the Resource with type "rt" = "oic.r.roles".**

Create	Read	Update	Delete	Notify
	get	post	delete	observe

## 6602 C.8 Security Profile

### 6603 C.8.1 Introduction

6604 Resource specifying supported and active security profile(s).

6605

### 6606 C.8.2 Well-known URI

6607 /oic/sec/sp

### 6608 C.8.3 Resource type

6609 The Resource Type is defined as: "oic.r.sp".

### 6610 C.8.4 OpenAPI 2.0 definition

```
6611 {
6612   "swagger": "2.0",
6613   "info": {
6614     "title": "Security Profile",
6615     "version": "v1.0-20190208",
6616     "license": {
6617       "name": "OCF Data Model License",
6618       "url":
6619 "https://github.com/openconnectivityfoundation/core/blob/e28a9e0a92e17042ba3e83661e4c0fbce8bdc4ba/LI
6620 CENSE.md",
6621       "x-copyright": "copyright 2016-2017, 2019 Open Connectivity Foundation, Inc. All rights
6622 reserved."
6623     },
6624     "termsOfService": "https://openconnectivityfoundation.github.io/core/DISCLAIMER.md"
6625   },
6626   "schemes": ["http"],
6627   "consumes": ["application/json"],
6628   "produces": ["application/json"],
6629   "paths": {
6630     "/oic/sec/sp" : {
6631       "get": {
6632         "description": "Resource specifying supported and active security profile(s).\n",
6633         "parameters": [
6634           { "$ref": "#/parameters/interface" }
6635         ],
6636         "responses": {
6637           "200": {
6638             "description": "",
6639             "x-example":
6640               {
6641                 "rt": ["oic.r.sp"],
6642                 "supportedprofiles" : ["1.3.6.1.4.1.51414.0.0.1.0", " 1.3.6.1.4.1.51414.0.0.2.0"],
6643                 "currentprofile" : "1.3.6.1.4.1.51414.0.0.1.0"
6644               },
6645             "schema": { "$ref": "#/definitions/SP" }
6646           },
6647           "400": {
6648             "description": "The request is invalid."
6649           }
6650         }
6651       },
6652       "post": {
6653         "description": "Sets or updates Device provisioning status data.\n",
6654         "parameters": [
6655           { "$ref": "#/parameters/interface" },
6656           {
6657             "name": "body",
6658             "in": "body",
6659             "required": true,
6660             "schema": { "$ref": "#/definitions/SP-Update" },
6661             "x-example":
6662               {
6663                 "supportedprofiles" : ["1.3.6.1.4.1.51414.0.0.1.0", " 1.3.6.1.4.1.51414.0.0.2.0"],
6664                 "currentprofile" : "1.3.6.1.4.1.51414.0.0.1.0"
6665               }
6666           }
6667         ],
6668         "responses": {
6669           "200": {
6670             "description": "",
6671             "x-example":
6672               {
6673                 "rt": ["oic.r.sp"],
6674                 "supportedprofiles" : ["1.3.6.1.4.1.51414.0.0.1.0", " 1.3.6.1.4.1.51414.0.0.2.0"],
6675                 "currentprofile" : "1.3.6.1.4.1.51414.0.0.1.0"
6676               }
6677         }
6678       }
6679     }
6680   }
6681 }
```

```

6676         },
6677         "schema": { "$ref": "#/definitions/SP" }
6678     },
6679     "400": {
6680         "description": "The request is invalid."
6681     }
6682 }
6683 }
6684 }
6685 },
6686 "parameters": {
6687     "interface" : {
6688         "in" : "query",
6689         "name" : "if",
6690         "type" : "string",
6691         "enum" : ["oic.if.baseline"]
6692     }
6693 },
6694 "definitions": {
6695     "SP" : {
6696         "properties": {
6697             "rt": {
6698                 "description": "Resource Type of the Resource.",
6699                 "items": {
6700                     "maxLength": 64,
6701                     "type": "string",
6702                     "enum": ["oic.r.sp"]
6703                 },
6704                 "minItems": 1,
6705                 "readOnly": true,
6706                 "type": "array"
6707             },
6708             "n": {
6709                 "$ref":
6710 "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
6711 schema.json#/definitions/n"
6712             },
6713             "id": {
6714                 "$ref":
6715 "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
6716 schema.json#/definitions/id"
6717             },
6718             "currentprofile": {
6719                 "description": "Security Profile currently active.",
6720                 "type": "string"
6721             },
6722             "supportedprofiles": {
6723                 "description": "Array of supported Security Profiles.",
6724                 "items": {
6725                     "type": "string"
6726                 },
6727                 "type": "array"
6728             },
6729             "if": {
6730                 "description": "The interface set supported by this Resource.",
6731                 "items": {
6732                     "enum": [
6733                         "oic.if.baseline"
6734                     ],
6735                     "type": "string"
6736                 },
6737                 "minItems": 1,
6738                 "readOnly": true,
6739                 "type": "array"
6740             }
6741         },
6742         "type" : "object",
6743         "required": ["supportedprofiles", "currentprofile"]
6744     },
6745     "SP-Update" : {
6746         "properties": {

```

```

6747     "currentprofile": {
6748         "description": "Security Profile currently active.",
6749         "type": "string"
6750     },
6751     "supportedprofiles": {
6752         "description": "Array of supported Security Profiles.",
6753         "items": {
6754             "type": "string"
6755         },
6756         "type": "array"
6757     }
6758 },
6759 "type" : "object"
6760 }
6761 }
6762 }
6763

```

6764 **C.8.5 Property definition**

6765 Table C-13 defines the Properties that are part of the "oic.r.sp" Resource Type.

6766 **Table C-13 – The Property definitions of the Resource with type "rt" = "oic.r.sp".**

Property name	Value type	Mandatory	Access mode	Description
rt	array: see schema	No	Read Only	Resource Type of the Resource.
n	multiple types: see schema	No	Read Write	
id	multiple types: see schema	No	Read Write	
currentprofile	string	Yes	Read Write	Security Profile currently active.
supportedprofiles	array: see schema	Yes	Read Write	Array of supported Security Profiles.
if	array: see schema	No	Read Only	The interface set supported by this Resource.
currentprofile	string		Read Write	Security Profile currently active.
supportedprofiles	array: see schema		Read Write	Array of supported Security Profiles.

6767 **C.8.6 CRUDN behaviour**

6768 Table C-14 defines the CRUDN operations that are supported on the "oic.r.sp" Resource Type.

6769 **Table C-14 – The CRUDN operations of the Resource with type "rt" = "oic.r.sp".**

Create	Read	Update	Delete	Notify
	get	post		observe

6770

6771  
6772  
6773  
6774

## Annex D (informative)

### OID definitions

6775 This annex captures the OIDs defined throughout the document. The OIDs listed are intended to  
6776 be used within the context of an X.509 v3 certificate. MAX is an upper bound for SEQUENCES of  
6777 UTF8Strings and OBJECT IDENTIFIERS and should not exceed 255.

```
6778 id-OCF OBJECT IDENTIFIER ::= { iso(1) identified-organization(3) dod(6) internet(1)
6779     private(4) enterprise(1) OCF(51414) }
6780
6781 -- OCF Security specific OIDs
6782
6783 id-ocfSecurity OBJECT IDENTIFIER ::= { id-OCF 0 }
6784 id-ocfX509Extensions OBJECT IDENTIFIER ::= { id-OCF 1 }
6785
6786 -- OCF Security Categories
6787
6788 id-ocfSecurityProfile ::= { id-ocfSecurity 0 }
6789 id-ocfCertificatePolicy ::= { id-ocfSecurity 1 }
6790
6791 -- OCF Security Profiles
6792
6793 sp-undefined ::= OBJECT IDENTIFIER { id-ocfSecurityProfile 0 }
6794 sp-baseline ::= OBJECT IDENTIFIER { id-ocfSecurityProfile 1 }
6795 sp-black ::= OBJECT IDENTIFIER { id-ocfSecurityProfile 2 }
6796 sp-blue ::= OBJECT IDENTIFIER { id-ocfSecurityProfile 3 }
6797 sp-purple ::= OBJECT IDENTIFIER { id-ocfSecurityProfile 4 }
6798
6799 sp-undefined-v0 ::= ocfSecurityProfileOID (id-sp-undefined 0)
6800 sp-baseline-v0 ::= ocfSecurityProfileOID {id-sp-baseline 0}
6801 sp-black-v0 ::= ocfSecurityProfileOID {id-sp-black 0}
6802 sp-blue-v0 ::= ocfSecurityProfileOID {id-sp-blue 0}
6803 sp-purple-v0 ::= ocfSecurityProfileOID {id-sp-purple 0}
6804
6805 ocfSecurityProfileOID ::= UTF8String
6806
6807 -- OCF Security Certificate Policies
6808
6809 ocfCertificatePolicy-v1 ::= { id-ocfCertificatePolicy 2}
6810
6811 -- OCF X.509v3 Extensions
6812
6813 id-ocfX509Extensions OBJECT IDENTIFIER ::= { id-OCF 1 }
6814 id-ocfCompliance OBJECT IDENTIFIER ::= { id-ocfX509Extensions 0 }
6815 id-ocfSecurityClaims OBJECT IDENTIFIER ::= { id-ocfX509Extensions 1 }
6816 id-ocfCPLAttributes OBJECT IDENTIFIER ::= { id-ocfX509Extensions 2 }
6817
6818 ocfVersion ::= SEQUENCE {
6819     major    INTEGER,
6820     minor    INTEGER,
6821     build    INTEGER}
6822
6823 ocfCompliance ::= SEQUENCE {
6824     version          ocfVersion,
6825     securityProfile SEQUENCE SIZE (1..MAX) OF ocfSecurityProfileOID,
6826     deviceName       UTF8String,
6827     deviceManufacturer UTF8String}
6828
6829 claim-secure-boot ::= ocfSecurityClaimsOID { id-ocfSecurityClaims 0 }
```

```
6830 claim-hw-backed-cred-storage ::= ocfSecurityClaimsOID { id-ocfSecurityClaims 1 }
6831
6832 ocfSecurityClaimsOID ::= OBJECT IDENTIFIER
6833
6834 ocfSecurityClaims ::= SEQUENCE SIZE (1..MAX) of ocfSecurityClaimsOID
6835
6836 cpl-at-IANAPen ::= OBJECT IDENTIFIER { id-ocfCPLAttributes 0 }
6837 cpl-at-model ::= OBJECT IDENTIFIER { id-ocfCPLAttributes 1 }
6838 cpl-at-version ::= OBJECT IDENTIFIER { id-ocfCPLAttributes 2 }
6839
6840 ocfCPLAttributes ::= SEQUENCE {
6841     cpl-at-IANAPen UTF8String,
6842     cpl-at-model UTF8String,
6843     cpl-at-version UTF8String}
```

6844  
6845  
6846  
6847

## Annex E (informative)

### Security considerations specific to Bridged Protocols

6848 The text in this Annex is provided for information only. This Annex has no normative impact. This  
6849 information is applicable at the time of initial publication and may become out of date.

#### 6850 E.1 Security Considerations specific to the AllJoyn Protocol

6851 This clause intentionally left empty.

#### 6852 E.2 Security Considerations specific to the Bluetooth LE Protocol

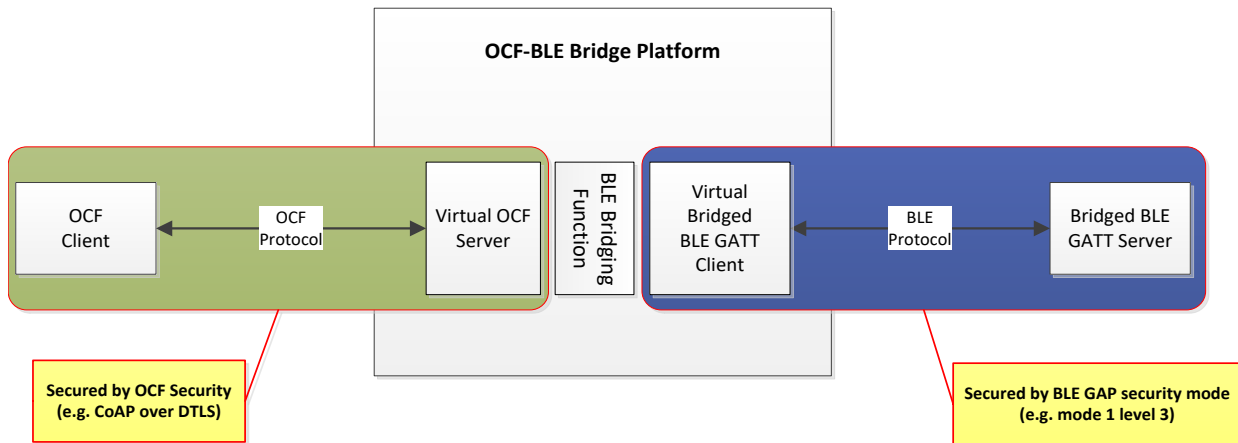
6853 BLE GAP supports two security modes, security mode 1 and security mode 2. Each security mode  
6854 has several security levels (see Table E.1)

6855 Security mode 1 and Security level 2 or higher would typically be considered secure from an OCF  
6856 perspective. The appropriate selection of security mode and level is left to the vendor.

6857 **Table E.1 GAP security mode**

GAP security mode	security level
Security mode 1	1 (no security)
	2 (Unauthenticated pairing with encryption)
	3 (Authenticated pairing with encryption)
	4 (Authenticated LE Secure Connections pairing with encryption)
Security mode 2	1 (Unauthenticated pairing with data signing)
	2 (Authenticated pairing with data signing)

6858 Figure E-1 shows how communications in both ecosystems of OCF-BLE Bridge Platform are  
6859 secured by their own security.



6860

6861 **Figure E-1 Security Considerations for BLE Bridge**

#### 6862 E.3 Security Considerations specific to the oneM2M Protocol

6863 This clause intentionally left empty.

6864 **E.4 Security Considerations specific to the U+ Protocol**

6865 A U+ server supports one of the TLS 1.2 cipher suites as in Table E.2 defined in IETF RFC 5246.

6866 **Table E.2 TLS 1.2 Cipher Suites used by U+**

Cipher Suite
TLS_RSA_WITH_AES_128_CBC_SHA256
TLS_RSA_WITH_AES_256_CBC_SHA256
TLS_RSA_WITH_AES_256_CCM
TLS_RSA_WITH_AES_256_CCM_8
TLS_RSA_WITH_AES_256_GCM_SHA384
TLS_DHE_RSA_WITH_AES_256_CBC_SHA256
TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA384
TLS_ECDH_ECDSA_WITH_AES_256_GCM_SHA384
TLS_ECDH_RSA_WITH_AES_256_CBC_SHA384
TLS_ECDH_RSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
TLS_ECDHE_ECDSA_WITH_AES_256_CCM
TLS_ECDHE_ECDSA_WITH_AES_256_CCM_8
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
TLS_DHE_RSA_WITH_AES_256_CCM
TLS_DHE_RSA_WITH_AES_256_CCM_8

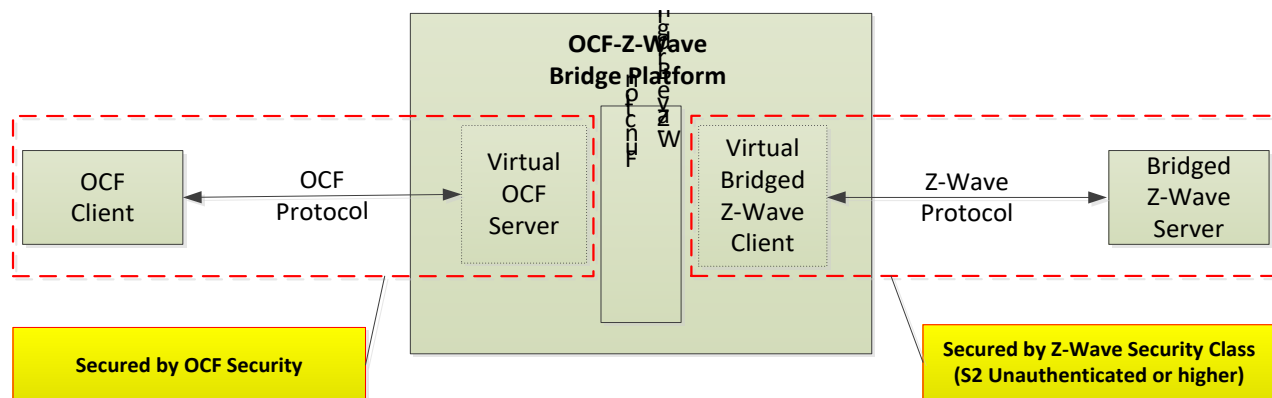
6867 The security of the Haier U+ Protocol is proprietary, and further details are presently unavailable.

6868 **E.5 Security Considerations specific to the Z-Wave Protocol**

6869 Z-Wave currently supports two kinds of security class which are S0 Security Class and S2 Security  
 6870 Class, as shown in Table E.3. Bridged Z-wave Servers using S2 Security Class for communication  
 6871 with a Virtual Bridged Client would typically be considered secure from an OCF perspective. The  
 6872 appropriate selection for S2 Security Class and Class Name is left to the vendor.

6873 Figure E-2 presents how OCF Client and Bridged Z-Wave Server communicate based upon their  
 6874 own security.





6875

6876

**Figure E-2 Security Considerations for Z-Wave Bridge**

6877 All 3 types of S2 Security Class such as S2 Access Control, S2 Authenticated and S2  
6878 Unauthenticated provides the following advantages from the security perspective;

- 6879 – The unique device specific key for every secure device enables validation of device identity and  
6880 prevents man-in-the-middle compromises to security
- 6881 – The Secure cryptographic key exchange methods during inclusion achieves high level of  
6882 security between the Virtual Z-Wave Client and the Bridged Z-Wave Server.
- 6883 – Out of band key exchange for product authentication which is combined with device specific  
6884 key prevents eavesdropping and man-in-the-middle attack vectors.

6885 See Table E.3 for a summary of Z-Wave Security Classes.

6886

**Table E.3 Z-Wave Security Class**

Security Class	Class Name	Validation of device identity	Key Exchange	Message Encapsulation
S2	S2 Access Control	Device Specific key	Out-of-band inclusion	Encrypted command transmission
	S2 Authenticated	Device Specific key	Out-of-band inclusion	Encrypted command transmission
	S2 Unauthenticated	Device Specific key	Z-wave RF band used for inclusion	Encrypted command transmission
S0	S0 Authenticated	N/A	Z-wave RF band used for inclusion	Encrypted command transmission

6887 On the other hand, S0 Security Class has the vulnerability of security during inclusion by  
6888 exchanging of temporary 'well-known key' (e.g. 1234). As a result of that, it could lead the  
6889 disclosure of the network key if the log of key exchange methods is captured, so Z-Wave devices  
6890 might be no longer secure in that case.

## 6891 E.6 Security Considerations specific to the Zigbee Protocol

6892 The Zigbee 3.0 stack supports multiple security levels. A security level is supported by both the  
6893 network (NWK) layer and application support (APS) layer. A security attribute in the Zigbee 3.0  
6894 stack, "nwkSecurityLevel", represents the security level of a device.

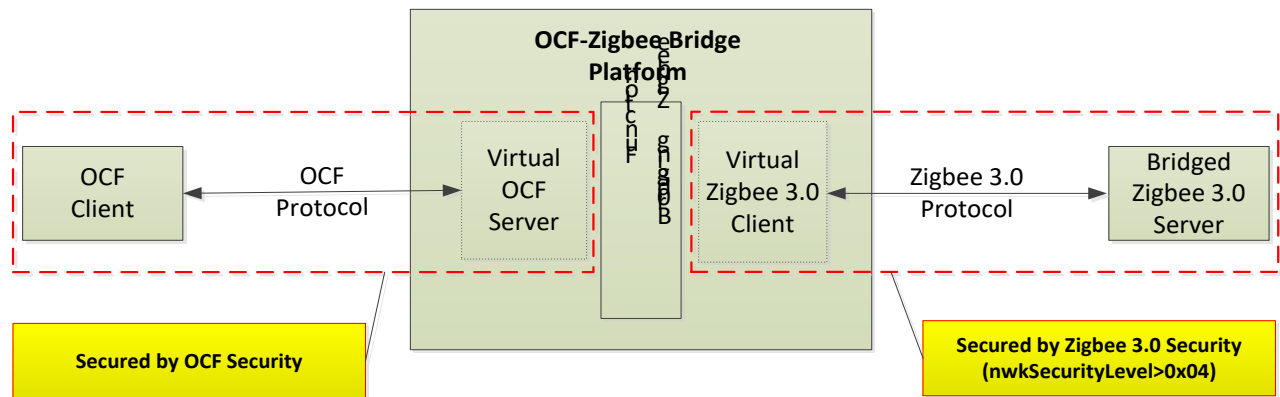
6895 The security level nwkSecurityLevel > 0x04 provides message integrity code (MIC) and/or AES128-  
 6896 CCM encryption (ENC). Zigbee Servers using nwkSecurityLevel > 0x04 would typically be  
 6897 considered secure from an OCF perspective. The appropriate selection for nwkSecurityLevel is left  
 6898 to the vendor.

6899 See Table E.4 for a summary of the Zigbee Security Levels.

6900 **Table E.4 Zigbee 3.0 Security Levels to the Network, and Application Support layers**

Security Level Identifier	Security Level Sub-Field	Security Attributes	Data Encryption	Frame Integrity (Length of M of MIC, in Number of Octets)
0x00	'000'	None	OFF	NO (M=0)
0x01	'001'	MIC-32	OFF	YES(M=4)
0x02	'010'	MIC-64	OFF	YES(M=8)
0x03	'011'	MIC-128	OFF	YES(M=16)
0x04	'100'	ENC	ON	NO(M=0)
0x05	'101'	ENC-MIC-32	ON	YES(M=4)
0x06	'110'	ENC-MIC-64	ON	YES(M=8)
0x07	'111'	ENC-MIC-128	ON	YES(M=16)

6901 Figure E-3 shows how communications in both ecosystems of OCF-Zigbee Bridge Platform are  
 6902 secured by their own security.



6903

6904 **Figure E-3 Security Considerations for Zigbee Bridge**

6905 **E.7 Security Considerations specific to the the EnOcean Radio Protocol**

6906 The EnOcean Radio Protocol supports four different security levels. The security level depends on  
 6907 which security mechanisms are used. Table E.5 defines them

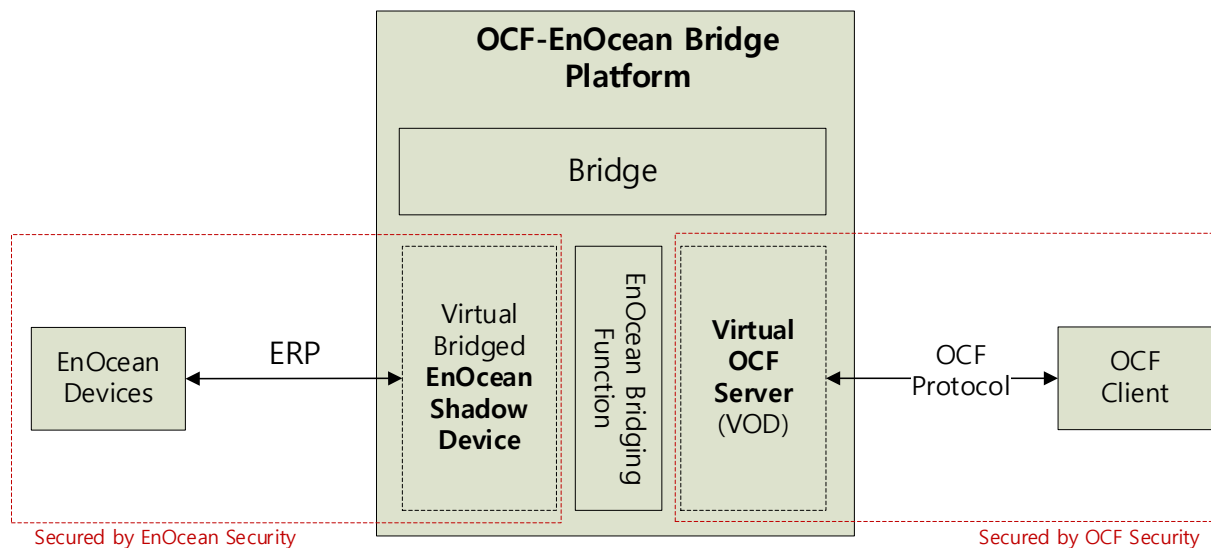
6908 **Table E.5 EnOcean Radio Protocol security levels**

Level	Features	Replay Attack Vulnerability	Eavesdropping Vulnerability
0	No Features (Unsecure)	Yes	Yes

1	With Encryption only	Yes	No
2	Without Encryption but with RLC and CMAC	No	Yes
3	With Encryption, RLC and CMAC	No	No

6909 The security levels 1 and 2 have been declared deprecated and shall not longer be used. Security  
6910 level 3 uses Variable AES Encryption, Rolling Code (RLC) and a cipher-based message  
6911 authentication code (CMAC) with private keys and public vectors. Technically each feature can be  
6912 combined with every other feature, even if it is obsolete or unreasonable.

6913 Figure E-4 shows how communications in both ecosystems of OCF- EnOcean Bridge Platform are  
6914 secured by their own security



6915

6916

**Figure E-4 Security Considerations for EnOcean Bridge**

6917