

OCF Security Specification

VERSION 2.1.0 | November 2019



OPEN CONNECTIVITY
FOUNDATION™

CONTACT admin@openconnectivity.org

Copyright Open Connectivity Foundation, Inc. © 2019.
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 © 2017-2019 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 10

24 4.1 Conventions..... 10

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.....	34
63	7.3.6	Manufacturer Certificate Based OTM.....	37
64	7.3.7	Vendor Specific OTMs.....	39
65	7.3.8	Establishing Owner Credentials.....	40
66	7.3.9	Security considerations regarding selecting an Ownership Transfer Method	
67		- Moved to OCF Onboarding Tool document.....	43
68	7.3.10	Security Profile Assignment.....	43
69	7.4	Provisioning.....	44
70	7.4.1	Provisioning Flows.....	44
71	7.5	Device Provisioning for OCF Cloud – moved to OCF Cloud Security document.....	46
72	8	Device Onboarding State Definitions.....	46
73	8.1	Device Onboarding General.....	46
74	8.2	Device Onboarding-Reset State Definition.....	48
75	8.3	Device Ready-for-OTM State Definition.....	48
76	8.4	Device Ready-for-Provisioning State Definition.....	48
77	8.5	Device Ready-for-Normal-Operation State Definition.....	49
78	8.6	Device Soft Reset State Definition.....	49
79	9	Security Credential Management.....	51
80	9.1	Preamble.....	51
81	9.2	Credential Lifecycle.....	51
82	9.2.1	Credential Lifecycle General.....	51
83	9.2.2	Creation.....	51
84	9.2.3	Deletion.....	51
85	9.2.4	Refresh.....	51
86	9.2.5	Revocation.....	51
87	9.3	Credential Types.....	52
88	9.3.1	Preamble.....	52
89	9.3.2	Pair-wise Symmetric Key Credentials.....	52
90	9.3.3	Group Symmetric Key Credentials.....	52
91	9.3.4	Asymmetric Authentication Key Credentials.....	53
92	9.3.5	Asymmetric Key Encryption Key Credentials.....	53
93	9.3.6	Certificate Credentials.....	53
94	9.3.7	Password Credentials.....	54
95	9.4	Certificate Based Key Management.....	54
96	9.4.1	Overview.....	54
97	9.4.2	X.509 Digital Certificate Profiles.....	55
98	9.4.3	Certificate Revocation List (CRL) Profile [Deprecated].....	63
99	9.4.4	Resource Model.....	63
100	9.4.5	Certificate Provisioning.....	64
101	9.4.6	CRL Provisioning [Deprecated].....	65
102	10	Device Authentication.....	66
103	10.1	Device Authentication General.....	66

104	10.2	Device Authentication with Symmetric Key Credentials	66
105	10.3	Device Authentication with Raw Asymmetric Key Credentials.....	66
106	10.4	Device Authentication with Certificates	66
107	10.4.1	Device Authentication with Certificates General.....	66
108	10.4.2	Role Assertion with Certificates	67
109	10.4.3	OCF PKI Roots	68
110	10.4.4	PKI Trust Store.....	68
111	10.4.5	Path Validation and extension processing.....	69
112	10.5	Device Authentication with OCF Cloud – moved to OCF Cloud Security	
113		document.....	70
114	11	Message Integrity and Confidentiality	71
115	11.1	Preamble	71
116	11.2	Session Protection with DTLS.....	71
117	11.2.1	DTLS Protection General.....	71
118	11.2.2	Unicast Session Semantics.....	71
119	11.2.3	Cloud Session Semantics – moved to OCF Cloud Security document	71
120	11.3	Cipher Suites	71
121	11.3.1	Cipher Suites General	71
122	11.3.2	Cipher Suites for Device Ownership Transfer	71
123	11.3.3	Cipher Suites for Symmetric Keys.....	72
124	11.3.4	Cipher Suites for Asymmetric Credentials.....	73
125	11.3.5	Cipher suites for OCF Cloud Credentials – moved to OCF Cloud Security	
126		document	73
127	12	Access Control	74
128	12.1	ACL Generation and Management	74
129	12.2	ACL Evaluation and Enforcement.....	74
130	12.2.1	ACL Evaluation and Enforcement General	74
131	12.2.2	Host Reference Matching	74
132	12.2.3	Resource Wildcard Matching	74
133	12.2.4	Multiple Criteria Matching	75
134	12.2.5	Subject Matching using Wildcards	75
135	12.2.6	Subject Matching using Roles.....	75
136	12.2.7	ACL Evaluation.....	76
137	13	Security Resources	78
138	13.1	Security Resources General	78
139	13.2	Device Owner Transfer Resource	79
140	13.2.1	Device Owner Transfer Resource General.....	79
141	13.2.2	OCF defined OTMs.....	82
142	13.3	Credential Resource	82
143	13.3.1	Credential Resource General.....	82
144	13.3.2	Properties of the Credential Resource	87
145	13.3.3	Key Formatting	89
146	13.3.4	Credential Refresh Method Details [Deprecated]	89
147	13.4	Certificate Revocation List	89

148	13.4.1	CRL Resource Definition [Deprecated]	89
149	13.5	ACL Resources	89
150	13.5.1	ACL Resources General	89
151	13.5.2	OCF Access Control List (ACL) BNF defines ACL structures.	90
152	13.5.3	ACL Resource	91
153	13.6	Access Manager ACL Resource [Deprecated]	96
154	13.7	Signed ACL Resource [Deprecated]	96
155	13.8	Provisioning Status Resource	96
156	13.9	Certificate Signing Request Resource	101
157	13.10	Roles Resource	102
158	13.11	Account Resource – moved to OCF Cloud Security document.....	103
159	13.12	Account Session Resource – moved to OCF Cloud Security document	103
160	13.13	Account Token Refresh Resource – moved to OCF Cloud Security document.....	103
161	13.14	Security Virtual Resources (SVRs) and Access Policy	103
162	13.15	SVRs, Discoverability and OCF Endpoints	103
163	13.16	Additional Privacy Consideration for Core Resources	104
164	13.17	Easy Setup Resource Device State.....	105
165	14	Security Hardening Guidelines/ Execution Environment Security	108
166	14.1	Preamble	108
167	14.2	Execution Environment Elements	108
168	14.2.1	Execution Environment Elements General	108
169	14.2.2	Secure Storage.....	108
170	14.2.3	Secure execution engine	111
171	14.2.4	Trusted input/output paths	111
172	14.2.5	Secure clock.....	111
173	14.2.6	Approved algorithms.....	111
174	14.2.7	Hardware tamper protection.....	112
175	14.3	Secure Boot.....	112
176	14.3.1	Concept of software module authentication.....	112
177	14.3.2	Secure Boot process	114
178	14.3.3	Robustness Requirements.....	114
179	14.4	Attestation	114
180	14.5	Software Update	114
181	14.5.1	Overview:	114
182	14.5.2	Recognition of Current Differences	115
183	14.5.3	Software Version Validation.....	116
184	14.5.4	Software Update.....	116
185	14.5.5	Recommended Usage.....	116
186	14.6	Non-OCF Endpoint interoperability.....	117
187	14.7	Security Levels	117
188	14.8	Security Profiles.....	118
189	14.8.1	Security Profiles General	118
190	14.8.2	Identification of Security Profiles (Normative)	118
191	14.8.3	Security Profiles	120

192	15	Device Type Specific Requirements.....	125
193	15.1	Bridging Security	125
194	15.1.1	Universal Requirements for Bridging to another Ecosystem	125
195	15.1.2	Additional Security Requirements specific to Bridged Protocols	126
196	Annex A	(informative) Access Control Examples.....	128
197	A.1	Example OCF ACL Resource	128
198	Annex B	(Informative) Execution Environment Security Profiles	129
199	Annex C	(normative) Resource Type definitions.....	130
200	C.1	List of Resource Type definitions	130
201	C.2	Access Control List-2	130
202	C.2.1	Introduction	130
203	C.2.2	Well-known URI	130
204	C.2.3	Resource type	130
205	C.2.4	OpenAPI 2.0 definition.....	130
206	C.2.5	Property definition	138
207	C.2.6	CRUDN behaviour	139
208	C.3	Credential	139
209	C.3.1	Introduction	139
210	C.3.2	Well-known URI	139
211	C.3.3	Resource type	139
212	C.3.4	OpenAPI 2.0 definition.....	139
213	C.3.5	Property definition	149
214	C.3.6	CRUDN behaviour	149
215	C.4	Certificate Signing Request.....	150
216	C.4.1	Introduction	150
217	C.4.2	Well-known URI	150
218	C.4.3	Resource type	150
219	C.4.4	OpenAPI 2.0 definition.....	150
220	C.4.5	Property definition	151
221	C.4.6	CRUDN behaviour	152
222	C.5	Device Owner Transfer Method.....	152
223	C.5.1	Introduction	152
224	C.5.2	Well-known URI	152
225	C.5.3	Resource type	152
226	C.5.4	OpenAPI 2.0 definition.....	152
227	C.5.5	Property definition	156
228	C.5.6	CRUDN behaviour	158
229	C.6	Device Provisioning Status	158
230	C.6.1	Introduction	158
231	C.6.2	Well-known URI	158
232	C.6.3	Resource type	158
233	C.6.4	OpenAPI 2.0 definition.....	158
234	C.6.5	Property definition	162
235	C.6.6	CRUDN behaviour	166

236	C.7	Asserted Roles	167
237	C.7.1	Introduction	167
238	C.7.2	Well-known URI	167
239	C.7.3	Resource type	167
240	C.7.4	OpenAPI 2.0 definition.....	167
241	C.7.5	Property definition	176
242	C.7.6	CRUDN behaviour	176
243	C.8	Security Profile	176
244	C.8.1	Introduction	176
245	C.8.2	Well-known URI	176
246	C.8.3	Resource type	177
247	C.8.4	OpenAPI 2.0 definition.....	177
248	C.8.5	Property definition	179
249	C.8.6	CRUDN behaviour	179
250	Annex D (informative)	OID definitions	180
251	Annex E (informative)	Security considerations specific to Bridged Protocols	182
252	E.1	Security Considerations specific to the AllJoyn Protocol	182
253	E.2	Security Considerations specific to the Bluetooth LE Protocol.....	182
254	E.3	Security Considerations specific to the oneM2M Protocol	182
255	E.4	Security Considerations specific to the U+ Protocol	183
256	E.5	Security Considerations specific to the Z-Wave Protocol.....	183
257	E.6	Security Considerations specific to the Zigbee Protocol	184
258			

259	FIGURES	
260	Figure 1 – OCF Interaction.....	10
261	Figure 2 – OCF Layers	12
262	Figure 3 – OCF Security Enforcement Points	14
263	Figure 4 – Use case-1 showing simple ACL enforcement.....	16
264	Figure 5 – Use case 2: A policy for the requested Resource is missing.....	17
265	Figure 6 – Example Resource definition with opaque Properties	18
266	Figure 7 – Property Level Access Control	18
267	Figure 8 – Onboarding Overview.....	20
268	Figure 9 – OCF Onboarding Process	22
269	Figure 10 – OCF's SRM Architecture	25
270	Figure 11 – Discover New Device Sequence.....	30
271	Figure 12 – A Just Works OTM	33
272	Figure 13 – Random PIN-based OTM	35
273	Figure 14 – Manufacturer Certificate Based OTM Sequence	38
274	Figure 15 – Vendor-specific Owner Transfer Sequence.....	40
275	Figure 16 – Symmetric Owner Credential Provisioning Sequence	42
276	Figure 17 – Example of Client-directed provisioning.....	45
277	Figure 18 – Device state model.....	47
278	Figure 19 – Client-directed Certificate Transfer.....	65
279	Figure 20 – Asserting a role with a certificate role credential.	68
280	Figure 21 – OCF Security Resources.....	78
281	Figure 22 – "/oic/sec/cred" Resource and Properties.....	79
282	Figure 23 – "/oic/sec/acl2" Resource and Properties.....	79
283	Figure 24 – Example of Soft AP and Easy Setup Resource in different Device states	105
284	Figure 25 – Software Module Authentication	113
285	Figure 26 – Verification Software Module.....	113
286	Figure 27 – Software Module Authenticity	114
287	Figure 28 – State transitioning diagram for software download	115
288	Figure A-1 – Example "/oic/sec/acl2" Resource.....	128
289	Figure E-1 Security Considerations for BLE Bridge	182
290	Figure E-2 Security Considerations for Z-Wave Bridge.....	184
291	Figure E-3 Security Considerations for Zigbee Bridge	185
292		

293 **Tables**

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

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

296 Table 3 – Random PIN-based OTM Details..... 35

297 Table 4 – Manufacturer Certificate Based OTM Details 38

298 Table 5 – Vendor-specific Owner Transfer Details 40

299 Table 6 – Symmetric Owner Credential Assignment Details 42

300 Table 7 – Steps describing Client -directed provisioning 45

301 Table 8 – X.509 v1 fields for Root CA Certificates..... 55

302 Table 9 - X.509 v3 extensions for Root CA Certificates 56

303 Table 10 - X.509 v1 fields for Intermediate CA Certificates 56

304 Table 11 – X.509 v3 extensions for Intermediate CA Certificates 56

305 Table 12 – X.509 v1 fields for End-Entity Certificates..... 57

306 Table 13 – X.509 v3 extensions for End-Entity Certificates 57

307 Table 14 – ACE2 Wildcard Matching Strings Description..... 74

308 Table 15 – Definition of the "/oic/sec/doxm" Resource 80

309 Table 16 – Properties of the "/oic/sec/doxm" Resource 80

310 Table 17 – Properties of the "oic.sec.didtype" type 81

311 Table 18 – Properties of the "oic.sec.doxmtype" type..... 82

312 Table 19 – Definition of the "/oic/sec/cred" Resource 83

313 Table 20 – Properties of the "/oic/sec/cred" Resource..... 84

314 Table 21 – Properties of the "oic.sec.creds" Property..... 85

315 Table 22: Properties of the "oic.sec.credusagetype" Property 86

316 Table 23 – Properties of the "oic.sec.pubdatatype" Property 86

317 Table 24 – Properties of the "oic.sec.privdatatype" Property 86

318 Table 25 – Properties of the "oic.sec.optdatatype" Property..... 87

319 Table 26 – Definition of the "oic.sec.roletype" type. 87

320 Table 27 – 128-bit symmetric key 89

321 Table 28 – 256-bit symmetric key 89

322 Table 29 – BNF Definition of OCF ACL 90

323 Table 30 – Value Definition of the "oic.sec.crudntype" Property 92

324 Table 31 – Definition of the "oic/sec/acl2" Resource 92

325 Table 32 – Properties of the "/oic/sec/acl2" Resource 93

326 Table 33 – "oic.sec.ace2" data type definition. 94

327 Table 34 – "oic.sec.ace2.resource-ref" data type definition. 94

328 Table 35 – Value definition "oic.sec.conntype" Property..... 94

329 Table 36 – Definition of the "/oic/sec/pstat" Resource 96

330 Table 37 – Properties of the "/oic/sec/pstat" Resource..... 97

331 Table 38 – Properties of the ".oic.sec.dostype" Property..... 98

332	Table 39 – Definition of the "oic.sec.dpmttype" Property	100
333	Table 40 – Value Definition of the "oic.sec.dpmttype" Property (Low-Byte)	100
334	Table 41 – Value Definition of the "oic.sec.dpmttype" Property (High-Byte).....	100
335	Table 42 – Definition of the "oic.sec.pomttype" Property	100
336	Table 43 – Value Definition of the "oic.sec.pomttype" Property	101
337	Table 44 – Definition of the "/oic/sec/csr" Resource	101
338	Table 45 – Properties of the "oic.r.csr" Resource	101
339	Table 46 – Definition of the "/oic/sec/roles" Resource	103
340	Table 47 – Properties of the "/oic/sec/roles" Resource.....	103
341	Table 48 – Core Resource Properties Access Modes given various Device States	104
342	Table 49 – Examples of Sensitive Data.....	109
343	Table 50 – Description of the software update bits.....	115
344	Table 51 – Definition of the "/oic/sec/sp" Resource	119
345	Table 52 – Properties of the "/oic/sec/sp" Resource.....	119
346	Table 53 – Dependencies of VOD Behaviour on Bridge state, as clarification of	
347	accompanying text.....	126
348	Table B.1 – OCF Security Profile	129
349	Table C.1 – Alphabetized list of security resources	130
350	Table C-1 – The Property definitions of the Resource with type "rt" = "oic.r.acl2".	138
351	Table C-2 – The CRUDN operations of the Resource with type "rt" = "oic.r.acl2".	139
352	Table C-3 – The Property definitions of the Resource with type "rt" = "oic.r.cred".	149
353	Table C-4 – The CRUDN operations of the Resource with type "rt" = "oic.r.cred".	150
354	Table C-5 – The Property definitions of the Resource with type "rt" = "oic.r.csr".....	152
355	Table C-6 – The CRUDN operations of the Resource with type "rt" = "oic.r.csr".	152
356	Table C-7 – The Property definitions of the Resource with type "rt" = "oic.r.doxm".	156
357	Table C-8 – The CRUDN operations of the Resource with type "rt" = "oic.r.doxm".	158
358	Table C-9 – The Property definitions of the Resource with type "rt" = "oic.r.pstat".	162
359	Table C-10 – The CRUDN operations of the Resource with type "rt" = "oic.r.pstat".	167
360	Table C-11 – The Property definitions of the Resource with type "rt" = "oic.r.roles".	176
361	Table C-12 – The CRUDN operations of the Resource with type "rt" = "oic.r.roles".	176
362	Table C-13 – The Property definitions of the Resource with type "rt" = "oic.r.sp".	179
363	Table C-14 – The CRUDN operations of the Resource with type "rt" = "oic.r.sp".	179
364	Table E.1 GAP security mode	182
365	Table E.2 TLS 1.2 Cipher Suites used by U+	183
366	Table E.3 Z-Wave Security Class.....	184
367	Table E.4 Zigbee 3.0 Security Levels to the Network, and Application Support layers	185
368		
369		

370 1 Scope

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

375 2 Normative References

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

379 ISO/IEC 30118-1:2018 Information technology -- Open Connectivity Foundation (OCF)
380 Specification -- Part 1: Core specification

381 <https://www.iso.org/standard/53238.html>

382 Latest version available at:

383 https://openconnectivity.org/specs/OCF_Core_Specification.pdf

384 ISO/IEC 30118-3:2018 Information technology -- Open Connectivity Foundation (OCF)
385 Specification -- Part 3: Bridging specification

386 <https://www.iso.org/standard/74240.html>

387 Latest version available at:

388 https://openconnectivity.org/specs/OCF_Bridging_Specification.pdf

389 OCF Wi-Fi Easy Setup, Information technology – Open Connectivity Foundation (OCF)
390 Specification – Part 7: Wi-Fi Easy Setup specification

391 Latest version available at:

392 https://openconnectivity.org/specs/OCF_Wi-Fi_Easy_Setup_Specification.pdf

393 OCF Cloud Specification, Information technology – Open Connectivity Foundation (OCF)
394 Specification – Part 8: Cloud Specification

395 Latest version available at:

396 https://openconnectivity.org/specs/OCF_Cloud_Specification.pdf

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

398 IETF RFC 2315, *PKCS #7: Cryptographic Message Syntax Version 1.5*, March 1998,

399 <https://tools.ietf.org/html/rfc2315>

400 IETF RFC 2898, *PKCS #5: Password-Based Cryptography Specification Version 2.0*, September
401 2000, <https://tools.ietf.org/html/rfc2898>

402 IETF RFC 2986, *PKCS #10: Certification Request Syntax Specification Version 1.7*, November
403 2000, <https://tools.ietf.org/html/rfc2986>

404 IETF RFC 4122, A Universally Unique IDentifier (UUID) URN Namespace, July 2005,

405 <https://tools.ietf.org/html/rfc4122>

406 IETF RFC 4279, *Pre-Shared Key Ciphersuites for Transport Layer Security (TLS)*, December
407 2005, <https://tools.ietf.org/html/rfc4279>

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

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

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

414 IETF RFC 5489, *ECDHE_PSK Cipher Suites for Transport Layer Security (TLS)*, March 2009,
415 <https://tools.ietf.org/html/rfc5489>

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

444

445 **3 Terms, definitions, and abbreviated terms**

446 **3.1 Terms and definitions**

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

449 ISO and IEC maintain terminological databases for use in standardization at the following
450 addresses:

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

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

453 **3.1.1**

454 **Access Management Service (AMS)**

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

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

458 **3.1.2**

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

460 **3.1.3**

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

462 **3.1.4**

463 **Client**

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

465 **3.1.5**

466 **Credential Management Service (CMS)**

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

469 **3.1.6**

470 **Device**

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

472 **3.1.7**

473 **Device Class**

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

476 **3.1.8**

477 **Device ID**

478 a stack instance identifier.

479 **3.1.9**

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

481 a logical entity that establishes device ownership

482 **3.1.10**

483 **3.1.11 Device Registration – moved to OCF Cloud Security document**

484 **End-Entity**

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

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

487 **3.1.12**

488 **Entity**

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

490 **3.1.13**
491 **OCF Interface**
492 Note 1 to entry: The details are defined in ISO/IEC 30118-1:2018.

493 **3.1.14**
494 **Intermediary**
495 a Device that implements both Client and Server roles and may perform protocol translation, virtual
496 device to physical device mapping or Resource translation

497 **3.1.15**
498 **OCF Cipher Suite**
499 a set of algorithms and parameters that define the cryptographic functionality of a Device. The OCF
500 Cipher Suite includes the definition of the public key group operations, signatures, and specific
501 hashing and encoding used to support the public key.

502 **3.1.16**
503 **OCF Cloud User – moved to OCF Cloud Security spec**

504 **3.1.17**
505 **OCF Rooted Certificate Chain**
506 a collection of X.509 v3 certificates in which each certificate chains to a trust anchor certificate
507 which has been issued by a certificate authority under the direction, authority, and approval of the
508 Open Connectivity Foundation Board of Directors as a trusted root for the OCF ecosystem.

509 **3.1.18**
510 **Onboarding Tool (OBT)**
511 a tool that implements DOTS(3.1.9), AMS(3.1.1) and CMS(3.1.5) functionality

512 **3.1.19**
513 **Out of Band Communication Channel**
514 any mechanism for delivery of a secret from one party to another, not specified by OCF

515 **3.1.20**
516 **Owner Credential (OC)**
517 a credential, provisioned to a Device, for the purposes of mutual authentication of the Device and
518 OBT(3.1.18) during subsequent interactions, identified by having a Subject UUID matching the
519 Resource Owner Id of the Device Ownership Transfer Resource hosted by a Device that has the
520 credential

521 **3.1.21**
522 **Platform ID**
523 Note 1 to entry: The details are defined in ISO/IEC 30118-1:2018.

524 **3.1.22**
525 **Property**
526 Note 1 to entry: The details are defined in ISO/IEC 30118-1:2018.

527 **3.1.23**
528 **Resource**
529 Note 1 to entry: The details are defined in ISO/IEC 30118-1:2018.

530 **3.1.24**
531 **Role (Network context)**
532 stereotyped behavior of a Device; one of [Client, Server or Intermediary]

533 **3.1.25**
534 **Role Identifier**
535 a Property of an OCF credentials Resource or element in a role certificate that identifies a privileged
536 role that a Server Device associates with a Client Device for the purposes of making authorization
537 decisions when the Client Device requests access to Device Resources.

538 **3.1.26**
539 **Secure Resource Manager (SRM)**
540 a module in the OCF Core that implements security functionality that includes management of
541 security Resources such as ACLs, credentials and Device owner transfer state.

542 **3.1.27**
543 **Security Virtual Resource (SVR)**
544 a resource supporting security features.

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

546 **3.1.28**
547 **Server**

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

549 **3.1.29**
550 **Trust Anchor**

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

553 **3.1.30**
554 **Unique Authenticable Identifier**

555 a unique identifier created from the hash of a public key and associated OCF Cipher Suite that is
556 used to create the Device ID.

557 Note 1 to entry: The ownership of a UAID may be authenticated by peer Devices.

558 **3.1.31**
559 **Device Configuration Resource (DCR)**

560 a Resource that is any of the following:

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

567 **3.1.32**
568 **Non-Configuration Resource (NCR)**

569 a Resource that is not a Device Configuration Resource (3.1.31).

570 **3.1.33**
571 **Bridged Device**

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

573 **3.1.34**
574 **Bridged Protocol**

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

576 **3.1.35**
577 **Bridge**
578 Note 1 to entry: The details are defined in ISO/IEC 30118-3:2018.

579 **3.1.36**
580 **Bridging Platform**
581 Note 1 to entry: The details are defined in ISO/IEC 30118-3:2018.

582 **3.1.37**
583 **Virtual Bridged Device**
584 Note 1 to entry: The details are defined in ISO/IEC 30118-3:2018.

585 **3.1.38**
586 **Virtual OCF Device**
587 Note 1 to entry: The details are defined in ISO/IEC 30118-3:2018.

588 **3.1.39**
589 **OCF Security Domain**
590 set of onboarded OCF Devices that are provisioned with credentialing information for confidential
591 communication with one another

592 **3.1.40**
593 **Owned (or "in Owned State")**
594 having the "owned" Property of the "/oic/sec/doxm" resource equal to "TRUE"

595 **3.1.41**
596 **Unowned (or "in Unowned State")**
597 having the "owned" Property of the "/oic/sec/doxm" resource equal to "FALSE"

598 **3.1.42 OCF Onboarding**
599 initial establishment of ownership over a Device, and initial provisioning of the Device for normal
600 operation

601 **3.2 Abbreviated terms**

602 **3.2.1**
603 **AC**
604 Access Control

605 **3.2.2**
606 **ACE**
607 Access Control Entry

608 **3.2.3**
609 **ACL**
610 Access Control List

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

615 **3.2.5**
616 **AMS**
617 Access Management Service

618 **3.2.6**
619 **CMS**
620 Credential Management Service

621 **3.2.7**
622 **CRUDN**
623 CREATE, RETREIVE, UPDATE, DELETE, NOTIFY

624 **3.2.8**
625 **CSR**
626 Certificate Signing Request

627 **3.2.9**
628 **CVC**
629 Code Verification Certificate

630 **3.2.10**
631 **ECC**
632 Elliptic Curve Cryptography

633 **3.2.11**
634 **ECDSA**
635 Elliptic Curve Digital Signature Algorithm

636 **3.2.12**
637 **EKU**
638 Extended Key Usage

639 **3.2.13**
640 **EPC**
641 Embedded Platform Credential

642 **3.2.14**
643 **EPK**
644 Embedded Public Key

645 **3.2.15**
646 **DOTS**
647 Device Ownership Transfer Service

648 **3.2.16**
649 **DPKP**
650 Dynamic Public Key Pair

651 **3.2.17**
652 **ID**
653 Identity/Identifier

654 **3.2.18**
655 **JSON**
656 JavaScript Object Notation.

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

658 **3.2.19**
659 **JWS**
660 JSON Web Signature.

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

662 **3.2.20**
663 **KDF**
664 Key Derivation Function

665 **3.2.21**
666 **MAC**
667 Message Authentication Code

668 **3.2.22**
669 **MITM**
670 Man-in-the-Middle

671 **3.2.23**
672 **NVRAM**
673 Non-Volatile Random-Access Memory

674 **3.2.24**
675 **OC**
676 Owner Credential

677 **3.2.25**
678 **OCSP**
679 Online Certificate Status Protocol

680 **3.2.26**
681 **OBT**
682 Onboarding Tool

683 **3.2.27**
684 **OID**
685 Object Identifier

686 **3.2.28**
687 **OTM**
688 Owner Transfer Method

689 **3.2.29**
690 **OWASP**
691 Open Web Application Security Project.

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

693 **3.2.30**
694 **PE**
695 Policy Engine

696 **3.2.31**
697 **PIN**
698 Personal Identification Number

699 **3.2.32**
700 **PPSK**
701 PIN-authenticated pre-shared key

702 **3.2.33**
703 **PRF**
704 Pseudo Random Function

705 **3.2.34**
706 **PSI**
707 Persistent Storage Interface

708 **3.2.35**
709 **PSK**
710 Pre Shared Key

711 **3.2.36**
712 **RBAC**
713 Role Based Access Control

714 **3.2.37**
715 **RM**
716 Resource Manager

717 **3.2.38**
718 **RNG**
719 Random Number Generator

720 **3.2.39**
721 **SBAC**
722 Subject Based Access Control

723 **3.2.40**
724 **SEE**
725 Secure Execution Environment

726 **3.2.41**
727 **SRM**
728 Secure Resource Manager

729 **3.2.42**
730 **SVR**
731 Security Virtual Resource

732 **3.2.43**
733 **SW**
734 Software

735 **3.2.44**
736 **UAID**
737 Unique Authenticable Identifier

738 **3.2.45**
739 **URI**
740 Uniform Resource Identifier

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

742 **3.2.46**
743 **VOD**
744 Virtual OCF Device

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

746 **3.2.47**
747 **RFNOP**
748 Ready for Normal

749 **3.2.48**
750 **RFOTM**
751 Ready for OTM

752 **3.2.49**
753 **RFPRO**
754 Ready for Provisioning

755 **3.2.50**
756 **SRESET**
757 Soft Reset

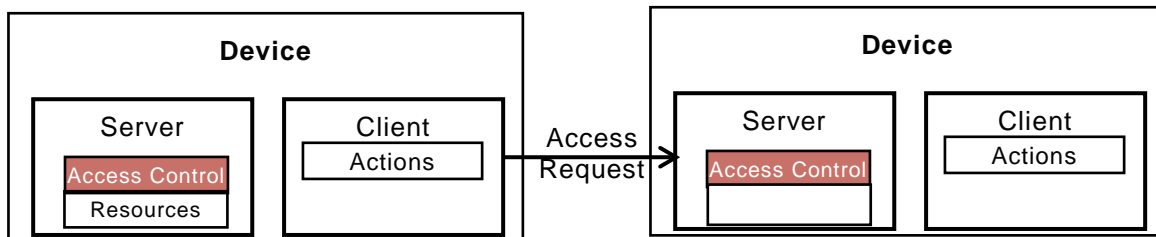
758 **4 Document Conventions and Organization**

759 **4.1 Conventions**

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

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

763 Figure 1 depicts interaction between OCF Devices.



764

765

Figure 1 – OCF Interaction

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

770 **4.2 Notation**

771 In this document, features are described as required, recommended, allowed or DEPRECATED as
772 follows:

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

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

777 **Recommended (or should).**

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

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

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

787 **Conditionally allowed (CA)**

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

790 **Conditionally required (CR)**

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

794 **DEPRECATED**

795 Although these features are still described in this document, they should not be implemented except
796 for backward compatibility. The occurrence of a deprecated feature during operation of an
797 implementation compliant with the current document has no effect on the implementation's
798 operation and does not produce any error conditions. Backward compatibility may require that a
799 feature is implemented and functions as specified but it shall never be used by implementations
800 compliant with this document.

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

802 Words that are emphasized are printed in italic.

803 **4.3 Data types**

804 See ISO/IEC 30118-1:2018.

805 **4.4 Document structure**

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

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

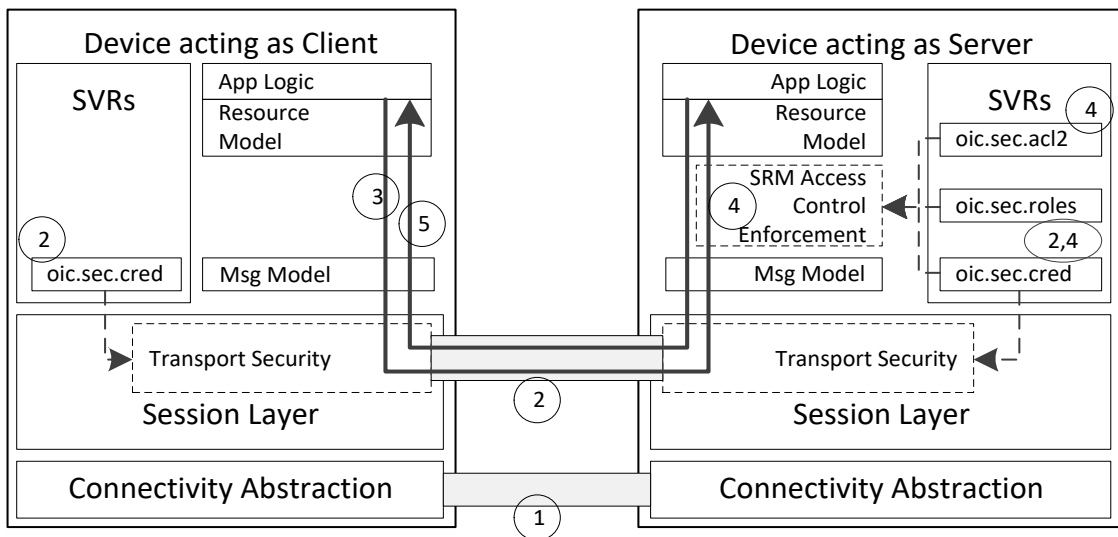
810

811 **5 Security Overview**

812 **5.1 Preamble**

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

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



821

822

Figure 2 – OCF Layers

- 823 1) The Client establishes a network connection to the Server (Device holding the Resources). The
 824 connectivity abstraction layer ensures the Devices are able to connect despite differences in
 825 connectivity options.
- 826 2) The Devices (e.g. Server and Client) exchange messages either with or without a mutually-
 827 authenticated secure channel between the two Devices.
- 828 a) The "/oic/sec/cred" Resource on each Devices holds the credentials used for mutual
 829 authentication and (when applicable) certificate validation.
- 830 b) Messages received over a secured channel are associated with a "deviceUUID". In the case
 831 of a certificate credential, the "deviceUUID" is in the certificate received from the other
 832 Device. In the case of a symmetric key credential, the "deviceUUID" is configured with the
 833 credential in the "/oic/sec/cred" Resource.
- 834 c) The Server can associate the Client with any number of roleid. In the case of mutual
 835 authentication using a certificate, the roleid (if any) are provided in role certificates; these
 836 are configured by the Client to the Server. In the case of a symmetric key, the allowed roleid
 837 (if any) are configured with the credential in the "/oic/sec/cred" Resource.

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

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

841 4) The Server receives the request.

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

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

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

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

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

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

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

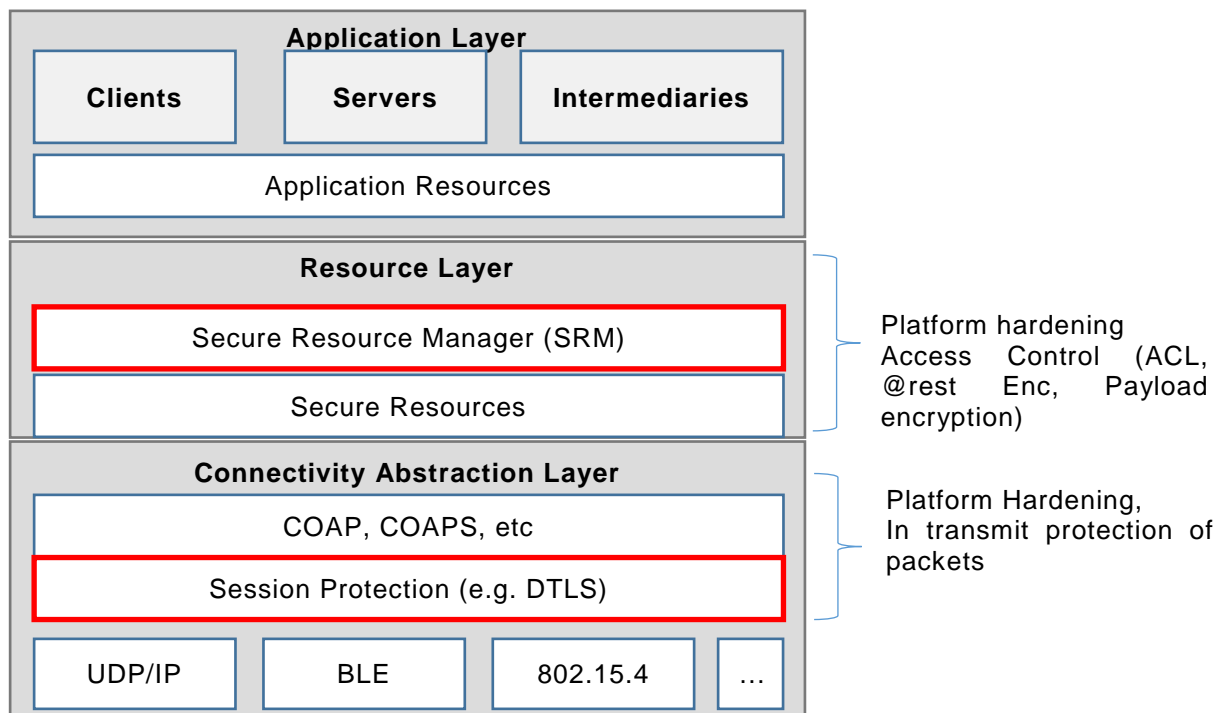
858 5) The Server sends a response back to the Client.

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

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

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

873 Figure 3 depicts OCF Security Enforcement Points.



874
875
876 **Figure 3 – OCF Security Enforcement Points**

877 **5.2 Access Control**

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

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

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

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

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

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

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

903 Example Wildcard Matching Policy:

```
904 "aclist2": [  
905   {  
906     "subject": {"conntype": "anon-clear" },  
907     "resources": [  
908       { "wc": "*" }  
909     ],  
910     "permission": 31  
911   },  
912   {  
913     "subject": {"conntype": "auth-crypt" },  
914     "resources": [  
915       { "wc": "*" }  
916     ],  
917     "permission": 31  
918   },  
919 ]
```

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

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

928 **5.2.1 ACL Architecture**

929 **5.2.1.1 ACL Architecture General**

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

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

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

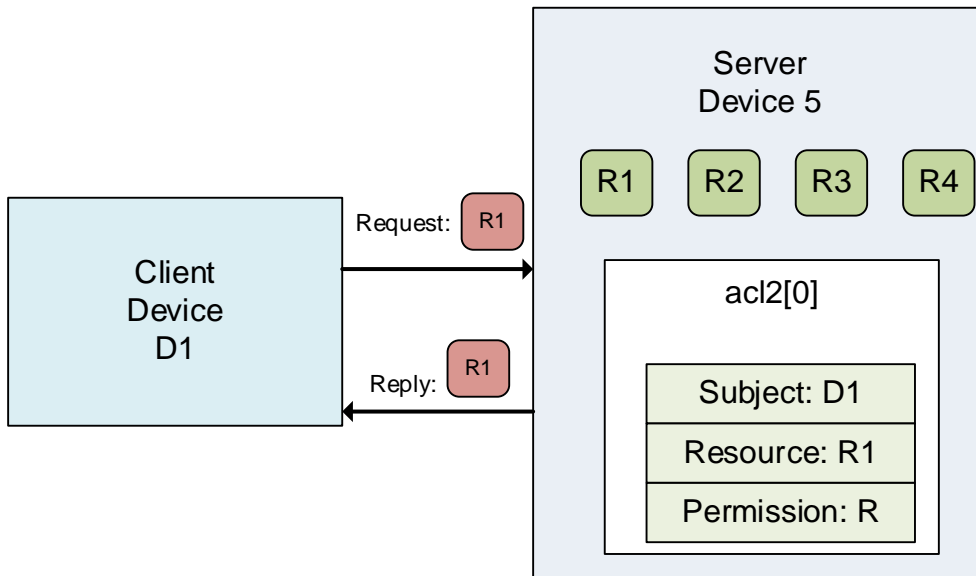
945 **5.2.1.2 Use of local ACLs**

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

948 The following use cases describe the operation of access control

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

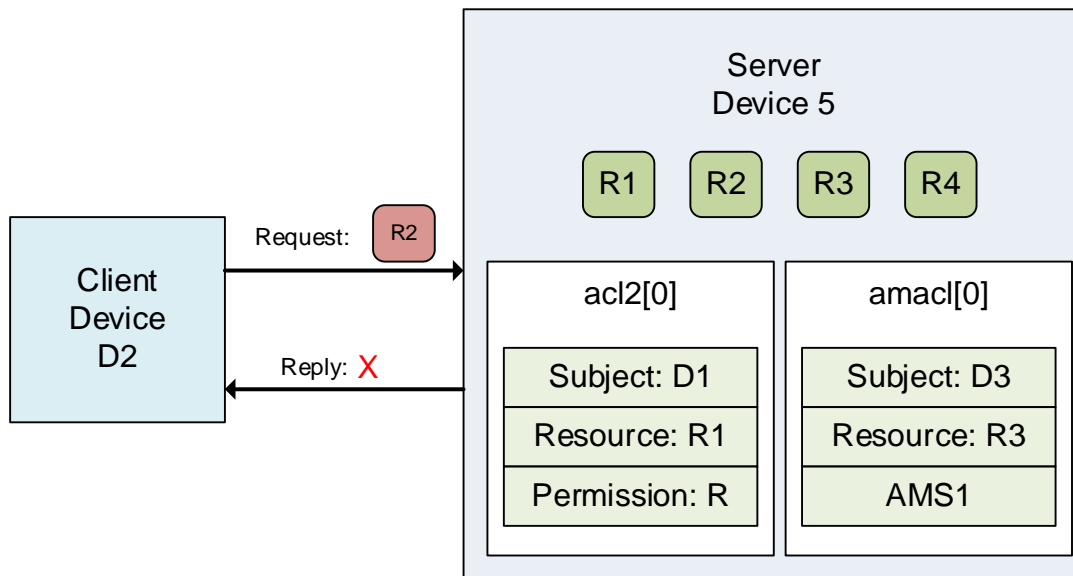
953



954

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

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



959

960

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

961 5.2.1.3 Use of AMS

962 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.
963

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

965 The AMS is authenticated by referencing a credential issued to the device identifier contained in
966 `"/oic/sec/acl2.owneruuid"`.

967 5.2.2 Access Control Scoping Levels

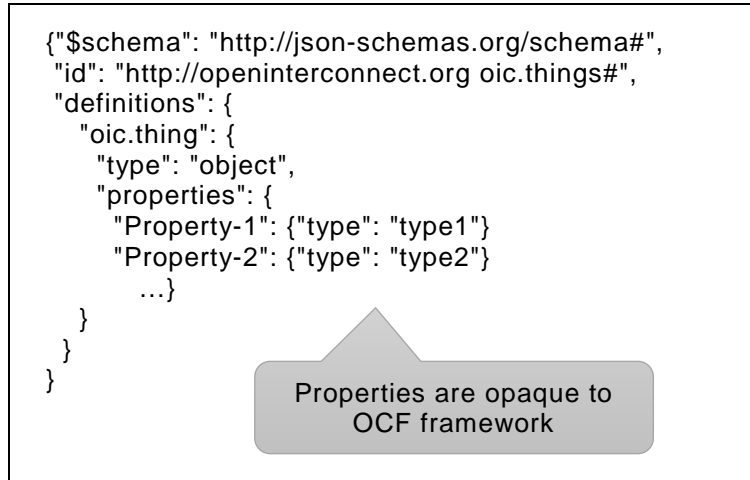
968 **Group Level Access** - Group scope means applying AC to the group of Devices that are grouped
969 for a specific context. Group Level Access means all group members have access to group data
970 but non-group members must be granted explicit access. Group level access is implemented using
971 Role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

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

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

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

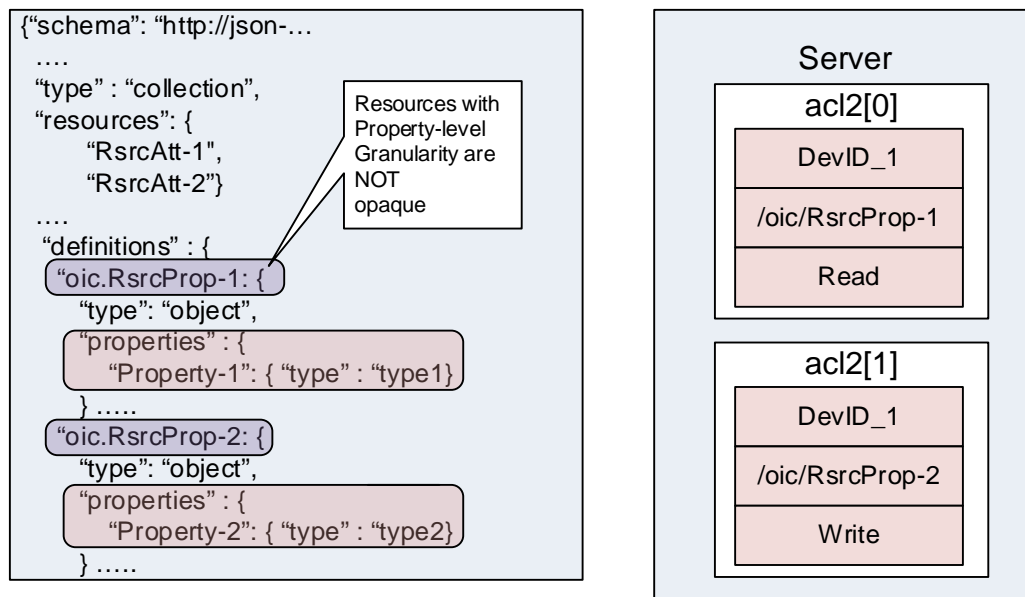
982 Controlling access to static Resources where it is impractical to redesign the Resource, it may
 983 appropriate to introduce a collection Resource that references the child Resources having separate
 984 access permissions. An example is shown Figure 6, where an "oic.thing" Resource has two
 985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



986

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

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



993

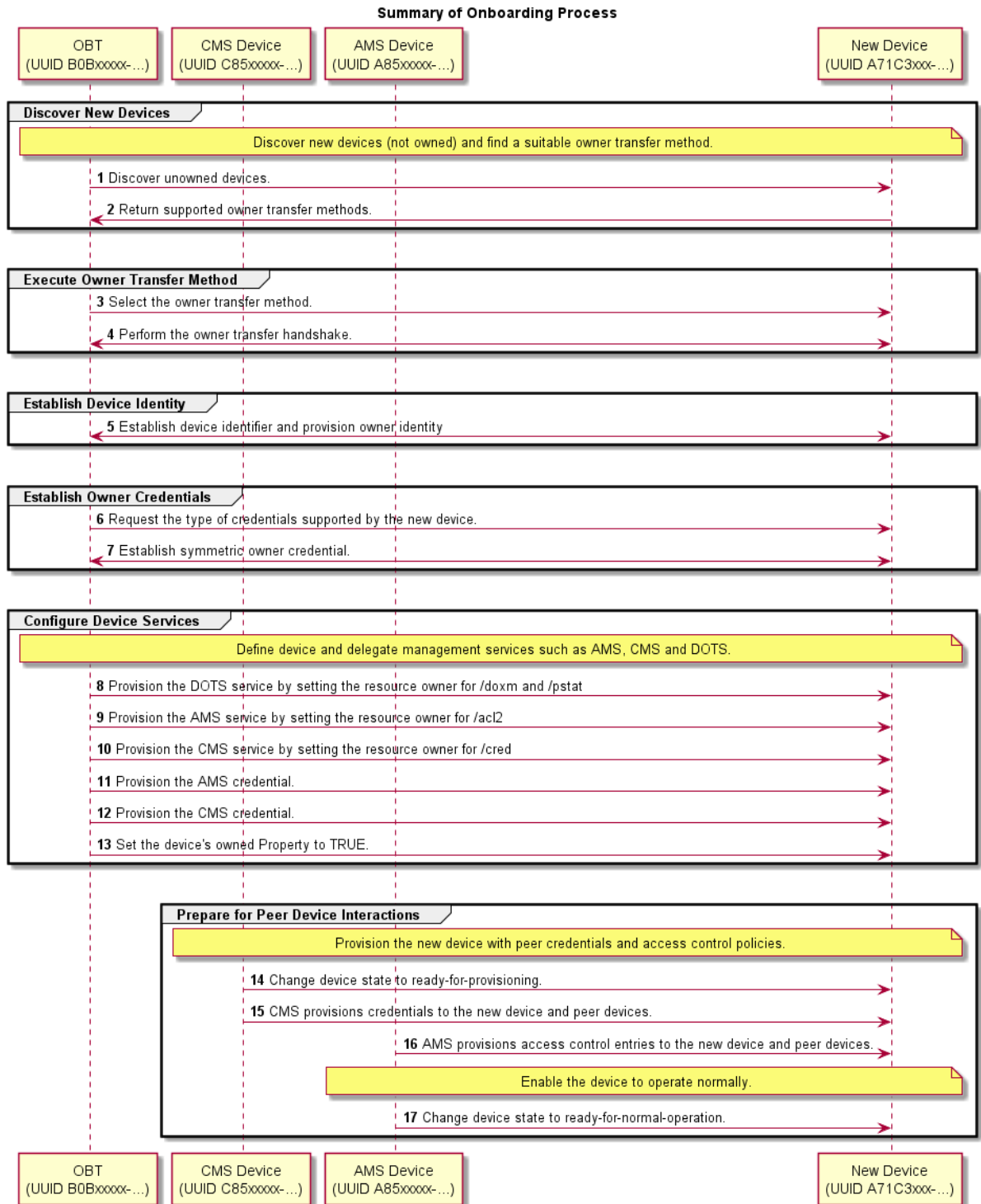
994 **Figure 7 – Property Level Access Control**

995 **5.3 Onboarding Overview**

996 **5.3.1 Onboarding General**

997 Before a Device becomes operational in an OCF environment and is able to interact with other
998 Devices, it needs to be appropriately onboarded. The first step in onboarding a Device is to
999 configure the ownership where the legitimate user that owns/purchases the Device uses an
1000 Onboarding tool (OBT) and using the OBT uses one of the Owner Transfer Methods (OTMs) to
1001 establish ownership. Once ownership is established, the OBT provisions the Device, at the end of
1002 which the Device becomes operational and is able to interact with other Devices in an OCF
1003 environment.

1004 Figure 8 depicts Onboarding Overview.



1005
1006

Figure 8 – Onboarding Overview

1007 This clause explains the onboarding and security provisioning process but leaves the provisioning
 1008 of non-security aspects to other OCF documents. In the context of security, all Devices are required
 1009 to be provisioned with minimal security configuration that allows the Device to securely
 1010 interact/communicate with other Devices in an OCF environment. This minimal security

1011 configuration is defined as the Onboarded Device "Ready for Normal Operation" and is specified
1012 in 7.5.

1013 Onboarding and provisioning implementations could utilize services defined outside this document,
1014 it is expected that in using other services, trust between the device being onboarded and the
1015 various tools is not transitive. This implies that the device being onboarded will individually
1016 authenticate the credentials of each and every tool used during the onboarding process; that the
1017 tools not share credentials or imply a trust relationship where one has not been established.

1018 **5.3.2 Onboarding Steps**

1019 The flowchart in Figure 9 shows the typical steps that are involved during onboarding. Although
1020 onboarding may include a variety of non-security related steps, the diagram focus is mainly on the
1021 security related configuration to allow a new Device to function within an OCF environment.
1022 Onboarding typically begins with the Device becoming an Owned Device followed by configuring
1023 the Device for the environment that it will operate in. This would include setting information such
1024 as who can access the Device and what actions can be performed as well as what permissions the
1025 Device has for interacting with other Devices.

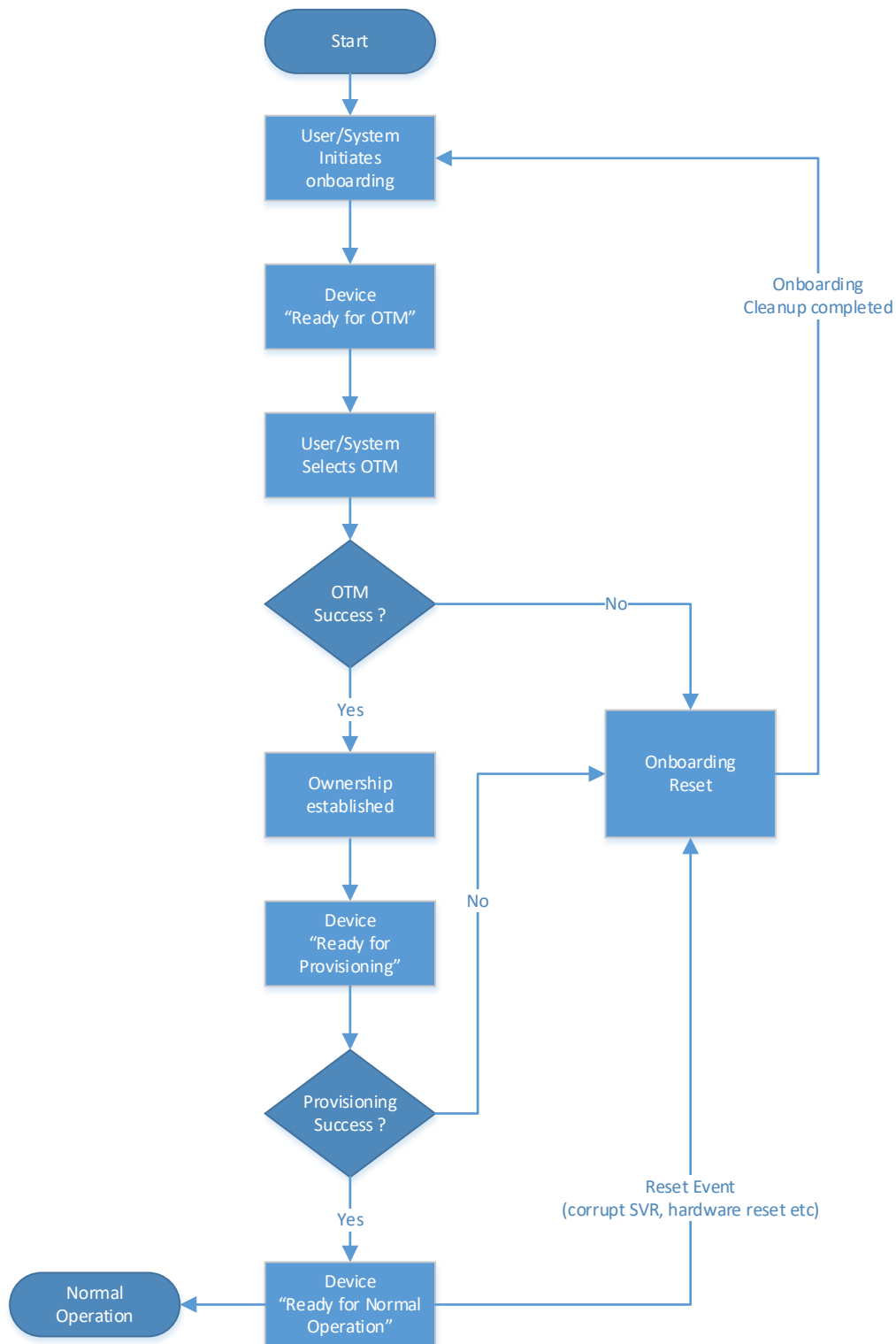


Figure 9 – OCF Onboarding Process

1026

1027

1028 **5.3.3 Establishing a Device Owner**

1029 The objective behind establishing Device ownership is to allow the legitimate user that
 1030 owns/purchased the Device to assert itself as the owner and manager of the Device. This is done
 Copyright Open Connectivity Foundation, Inc. © 2016-2019. All rights Reserved 22

1031 through the use of a DOTS that includes the creation of an ownership context between the new
1032 Device and the DOTS and asserts operational control and management of the Device. The DOTS
1033 is hosted on an OBT.

1034 The DOTS uses one of the OTMs specified in 7.3 to securely establish Device ownership.

1035 An OTM establishes a new owner (the operator of DOTS) that is authorized to manage the Device.
1036 Owner transfer establishes the following

1037 – The DOTS provisions an Owner Credential (OC) to the "creds" Property in the "/oic/sec/cred"
1038 Resource of the Device. This OC allows the Device and DOTS to mutually authenticate during
1039 subsequent interactions. The OC associates the DOTS Device UUID with the "rowneruid"
1040 Property of the "/oic/sec/doxm" Resource establishing it as the resource owner.

1041 – The Device owner establishes trust in the Device through the OTM.

1042 – Preparing the Device for provisioning by providing credentials that may be needed.

1043 **5.3.4 Provisioning for Normal Operation**

1044 Once the Device has the necessary information to initiate provisioning, the next step is to provision
1045 additional security configuration that allows the Device to become operational. This can include
1046 setting various parameters and may also involve multiple steps. Also provisioning of ACL's for the
1047 various Resources hosted by the Server on the Device is done at this time. The provisioning step
1048 is not limited to this stage only. Device provisioning can happen at multiple stages in the Device's
1049 operational lifecycle. However specific security related provisioning of Resource and Property state
1050 would likely happen at this stage at the end of which, each Device reaches the Onboarded Device
1051 "Ready for Normal Operation" State. The "Ready for Normal Operation" State is expected to be
1052 consistent and well defined regardless of the specific OTM used or regardless of the variability in
1053 what gets provisioned. However individual OTM mechanisms and provisioning steps may specify
1054 additional configuration of Resources and Property states. The minimal mandatory configuration
1055 required for a Device to be in "Ready for Normal Operation" state is specified in 8.

1056 **5.3.5 Device Provisioning for OCF Cloud and Device Registration Overview – moved to** 1057 **OCF Cloud Security document**

1058 This clause is intentionally left blank.

1059 **5.3.6 OCF Compliance Management System**

1060 The OCF Compliance Management System (OCMS) is a service maintained by the OCF that
1061 provides Certification status and information for OCF Devices.

1062 The OCMS shall provide a JSON-formatted Certified Product List (CPL), hosted at the URI:
1063 <https://www.openconnectivity.org/certification/ocms-cpl.json>

1064 The OBT shall possess the Root Certificate needed to enable https connection to the URI
1065 <https://www.openconnectivity.org/certification/ocms-cpl.json>.

1066 The OBT should periodically refresh its copy of the CPL via the URI
1067 <https://www.openconnectivity.org/certification/ocms-cpl.json>, as appropriate to OCF Security
1068 Domain owner policy requirements.

1069 **5.4 Provisioning**

1070 **5.4.1 Provisioning General**

1071 In general, provisioning may include processes during manufacturing and distribution of the Device
1072 as well as processes after the Device has been brought into its intended environment (parts of
1073 onboarding process). In this document, security provisioning includes, processes after ownership
1074 transfer (even though some activities during ownership transfer and onboarding may lead to
1075 provisioning of some data in the Device) configuration of credentials for interacting with
Copyright Open Connectivity Foundation, Inc. © 2016-2019. All rights Reserved 23

1076 provisioning services, configuration of any security related Resources and credentials for dealing
1077 with any services that the Device need to contact later on.

1078 Once the ownership transfer is complete, the Device needs to engage with the CMS and AMS to
1079 be provisioned with proper security credentials and parameters for regular operation. These
1080 parameters can include:

- 1081 – Security credentials through a CMS, currently assumed to be deployed in the same OBT.
- 1082 – Access control policies and ACLs through an AMS, currently assumed to be deployed in the
1083 same OBT, but may be part of AMS in future.

1084 Devices are aware of their security provisioning status. Self-awareness allows them to be proactive
1085 about provisioning or re-provisioning security Resources as needed to achieve the devices
1086 operational goals.

1087 **5.4.2 Provisioning other services**

1088 To be able to support the use of potentially different device management service hosts, each Device
1089 Secure Virtual Resource (SVR) has an associated Resource owner identified in the Resource's
1090 rowneruuid Property.

1091 The "rowneruuid" Property of the "/oic/sec/doxm" and "/oic/sec/pstat" resources identifies the
1092 DOTS.

1093 The "rowneruuid" Property of the "/oic/sec/cred" resource identifies the CMS.

1094 The "rowneruuid" Property of the "/oic/sec/acl2" resource identifies the AMS.

1095 The DOTS provisions credentials that enable secure connections between OCF Services and the
1096 new Device. The DOTS initiates client-directed provisioning by signaling the OCF Service.

1097 **5.4.3 Provisioning Credentials for Normal Operation**

1098 The "/oic/sec/cred" Resource supports multiple types of credentials including:

- 1099 – Pairwise symmetric keys
- 1100 – Group symmetric keys
- 1101 – Certificates
- 1102 – Raw asymmetric keys

1103 The CMS securely provisions credentials for Device-to-Device interactions using the CMS
1104 credential provisioned by the DOTS.

1105 The following example describes how a Device updates a symmetric key credential involving a peer
1106 Device. The Device discovers the credential to be updated; for example, a secure connection
1107 attempt fails. The CMS returns an updated symmetric key credential. The CMS updates the
1108 corresponding symmetric key credential on the peer Device.

1109 **5.4.4 Role Assignment and Provisioning for Normal Operation**

1110 The Servers, receiving requests for Resources they host, need to verify the role identifier(s)
1111 asserted by the Client requesting the Resource and compare that role identifier(s) with the
1112 constraints described in the Server's ACLs Thus, a Client Device may need to be provisioned with
1113 one or more role credentials.

1114 Each Device holds the role information as a Property within the credential Resource.

1115 Once provisioned, the Client can assert the role it is using as described in 10.4.2, if it has a
1116 certificate role credential.

1117 All provisioned roles are used in ACL enforcement. When a server has multiple roles provisioned
1118 for a client, access to a Resource is granted if it would be granted under any of the roles.

1119 5.4.5 ACL provisioning

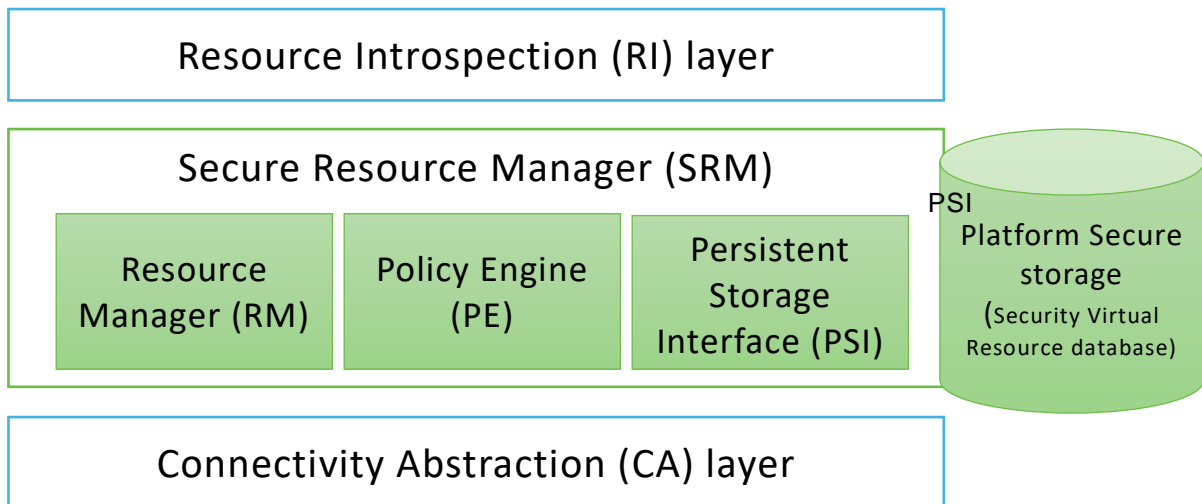
1120 ACL provisioning is performed over a secure connection between the AMS and its Devices. The
1121 AMS provisions the ACL by updating the Device's ACL Resource.

1122 5.5 Secure Resource Manager (SRM)

1123 SRM plays a key role in the overall security operation. In short, SRM performs both management
1124 of SVR and access control for requests to access and manipulate Resources. SRM consists of 3
1125 main functional elements:

- 1126 – A Resource manager (RM): responsible for 1) Loading SVRs from persistent storage (using PSI)
1127 as needed. 2) Supplying the Policy Engine (PE) with Resources upon request. 3) Responding
1128 to requests for SVRs. While the SVRs are in SRM memory, the SVRs are in a format that is
1129 consistent with device-specific data store format. However, the RM will use JSON format to
1130 marshal SVR data structures before being passed to PSI for storage, or travel off-device.
- 1131 – A Policy Engine (PE) that takes requests for access to SVRs and based on access control
1132 policies responds to the requests with either "ACCESS_GRANTED" or "ACCESS_DENIED". To
1133 make the access decisions, the PE consults the appropriate ACL and looks for best Access
1134 Control Entry (ACE) that can serve the request given the subject (Device or role) that was
1135 authenticated by DTLS.
- 1136 – Persistent Storage Interface (PSI): PSI provides a set of APIs for the RM to manipulate files in
1137 its own memory and storage. The SRM design is modular such that it may be implemented in
1138 the Platform's secure execution environment; if available.

1139 Figure 10 depicts OCF's SRM Architecture.



1140

1141 **Figure 10 – OCF's SRM Architecture**

1142 5.6 Credential Overview

1143 Devices may use credentials to prove the identity and role(s) of the parties in bidirectional
1144 communication. Credentials can be symmetric or asymmetric. Each device stores secret and public

1145 parts of its own credentials where applicable, as well as credentials for other devices that have
1146 been provided by the DOTS or a CMS. These credentials are then used in the establishment of
1147 secure communication sessions (e.g. using DTLS) to validate the identities of the participating
1148 parties. Role credentials are used once an authenticated session is established, to assert one or
1149 more roles for a device.

1150

1151 **6 Security for the Discovery Process**

1152 **6.1 Preamble**

1153 The main function of a discovery mechanism is to provide Universal Resource Identifiers (URIs,
1154 called links) for the Resources hosted by the Server, complemented by attributes about those
1155 Resources and possible further link relations. (in accordance to clause 10 in ISO/IEC 30118-1:2018)

1156 **6.2 Security Considerations for Discovery**

1157 When defining discovery process, care must be taken that only a minimum set of Resources are
1158 exposed to the discovering entity without violating security of sensitive information or privacy
1159 requirements of the application at hand. This includes both data included in the Resources, as well
1160 as the corresponding metadata.

1161 To achieve extensibility and scalability, this document does not provide a mandate on
1162 discoverability of each individual Resource. Instead, the Server holding the Resource will rely on
1163 ACLs for each Resource to determine if the requester (the Client) is authorized to see/handle any
1164 of the Resources.

1165 The `"/oic/sec/acl2"` Resource contains ACL entries governing access to the Server hosted
1166 Resources. (See 13.5)

1167 Aside from the privacy and discoverability of Resources from ACL point of view, the discovery
1168 process itself needs to be secured. This document sets the following requirements for the discovery
1169 process:

1170 1) Providing integrity protection for discovered Resources.

1171 2) Providing confidentiality protection for discovered Resources that are considered sensitive.

1172 The discovery of Resources is done by doing a RETRIEVE operation (either unicast or multicast)
1173 on the known `"/oic/res"` Resource.

1174 The discovery request is sent over a non-secure channel (multicast or unicast without DTLS), a
1175 Server cannot determine the identity of the requester. In such cases, a Server that wants to
1176 authenticate the Client before responding can list the secure discovery URI (e.g.
1177 `coaps://IP:PORT/oic/res`) in the unsecured `"/oic/res"` Resource response. This means the secure
1178 discovery URI is by default discoverable by any Client. The Client will then be required to send a
1179 separate unicast request using DTLS to the secure discovery URI.

1180 For example, a Client with Device Id `"d1"` (UUID:`"0685B960-736F-46F7-BEC0-9E6CBD61ADC1"`)
1181 makes a RETRIEVE request on the `"/door"` Resource hosted on a Server with Device Id `"d3"` where
1182 `d3` has the ACL2s:

```
1183 {  
1184   "aclist2": [  
1185     {  
1186       "subject": {"uuid": "0685B960-736F-46F7-BEC0-9E6CBD61ADC1"},  
1187       "resources": [{"href": "/door"}],  
1188       "permission": 2, // RETRIEVE  
1189       "aceid": 1  
1190     },  
1191     {  
1192       "subject": {"authority": "owner", "role": "owner"},  
1193       "resources": [{"href": "/door"}],  
1194       "permission": 2, // RETRIEVE
```

```

1195     "aceid": 2
1196   },
1197   {
1198     "subject": {"uuid": "0685B960-736F-46F7-BEC0-9E6CBD61ADC1"},
1199     "resources": [{"href": "/door/lock"}],
1200     "permission": 4, // UPDATE
1201     "aceid": 3
1202   }
1203 ],
1204 "rowneruuid": "0685B960-736F-46F7-BEC0-9E6CBD61ADC1"
1205 }

```

1206 The ACL indicates that Client "d1" has RETRIEVE permissions on the Resource. Hence when
1207 device "d1" does a discovery on the "/door" Resource of the Server "d3", the response will include
1208 all the URIs in the "/door" Resource. Client "d2" without a Role ID "owner" will get an error response
1209 that includes no URI.

1210 Discovery results delivered to d1 regarding d3's "/door" Resource from the secure interface:

```

1211 [
1212   {
1213     "href": "/door",
1214     "rel": "self",
1215     "rt": ["oic.wk.col"],
1216     "if": ["oic.if.ll", "oic.if.b", "oic.if.baseline"],
1217     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:5555"}]
1218   },
1219   {
1220     "href": "/door/lock",
1221     "rt": ["oic.r.lock.status"],
1222     "if": ["oic.if.a", "oic.if.baseline"],
1223     "eps": [{"ep": "coaps://[2001:db8:a::b1d4]:5555"}]
1224   }
1225 ]

```

1226 **7 Security Provisioning**

1227 **7.1 Device Identity**

1228 **7.1.1 General Device Identity**

1229 Each Device, which is a logical device, is identified with a Device ID.

1230 Devices shall be identified by a Device ID value that is established as part of device onboarding.
1231 The "/oic/sec/doxm" Resource specifies the Device ID format (e.g. "urn:uuid"). Device IDs shall be
1232 unique within the scope of operation of the corresponding OCF Security Domain, and should be
1233 universally unique. The DOTS shall ensure Device ID of the new Device is unique within the scope
1234 of the owner's OCF Security Domain. The DOTS shall verify the chosen new device identifier does
1235 not conflict with Device IDs previously introduced into the OCF Security Domain.

1236 Devices maintain an association of Device ID and cryptographic credential using a "/oic/sec/cred"
1237 Resource. Devices regard the "/oic/sec/cred" Resource as authoritative when verifying
1238 authentication credentials of a peer device.

1239 A Device maintains its Device ID in the "/oic/sec/doxm" Resource. It maintains a list of credentials,
1240 both its own and other Device credentials, in the "/oic/sec/cred" Resource. The device ID can be
1241 used to distinguish between a device's own credential, and credentials for other devices.
1242 Furthermore, the "/oic/sec/cred" Resource may contain multiple credentials for the device.

1243 When using manufacturer certificates, the certificate should bind the ID to the stored secret in the
1244 device as described later in this clause.

1245 A physical Device, referred to as a Platform in OCF documents, may host multiple Devices. The
1246 Platform is identified by a Platform ID. The Platform ID shall be globally unique and inserted in the
1247 device in an integrity protected manner (e.g. inside secure storage or signed and verified).

1248 An OCF Platform may have a secure execution environment, which shall be used to secure unique
1249 identifiers and secrets. If a Platform hosts multiple devices, some mechanism is needed to provide
1250 each Device with the appropriate and separate security.

1251 **7.1.2 Device Identity for Devices with UAID [Deprecated]**

1252 This clause is intentionally left blank.

1253 **7.2 Device Ownership**

1254 This is an informative clause. Devices are logical entities that are security endpoints that have an
1255 identity that is authenticable using cryptographic credentials. A Device is Unowned when it is first
1256 initialized. Establishing device ownership is a process by which the device asserts its identity to
1257 the DOTS and the DOTS provisions an owner identity. This exchange results in the device changing
1258 its ownership state, thereby preventing a different DOTS from asserting administrative control over
1259 the device.

1260 The ownership transfer process starts with the OBT discovering a new device that is in Unowned
1261 state through examination of the "Owned" Property of the "/oic/sec/doxm" Resource of the new
1262 device. At the end of ownership transfer, the following is accomplished:

- 1263 1) The DOTS establishes a secure session with new device.
- 1264 2) Optionally asserts any of the following:
 - 1265 a) Proximity (using PIN) of the OBT to the Platform.
 - 1266 b) Manufacturer's certificate asserting Platform vendor, model and other Platform specific
1267 attributes.
- 1268 3) Determines the device identifier.

- 1269 4) Determines the device owner.
- 1270 5) Specifies the device owner (e.g. Device ID of the OBT).
- 1271 6) Provisions the device with owner's credentials.
- 1272 7) Sets the "Owned" state of the new device to TRUE.

1273 .

1274 **7.3 Device Ownership Transfer Methods**

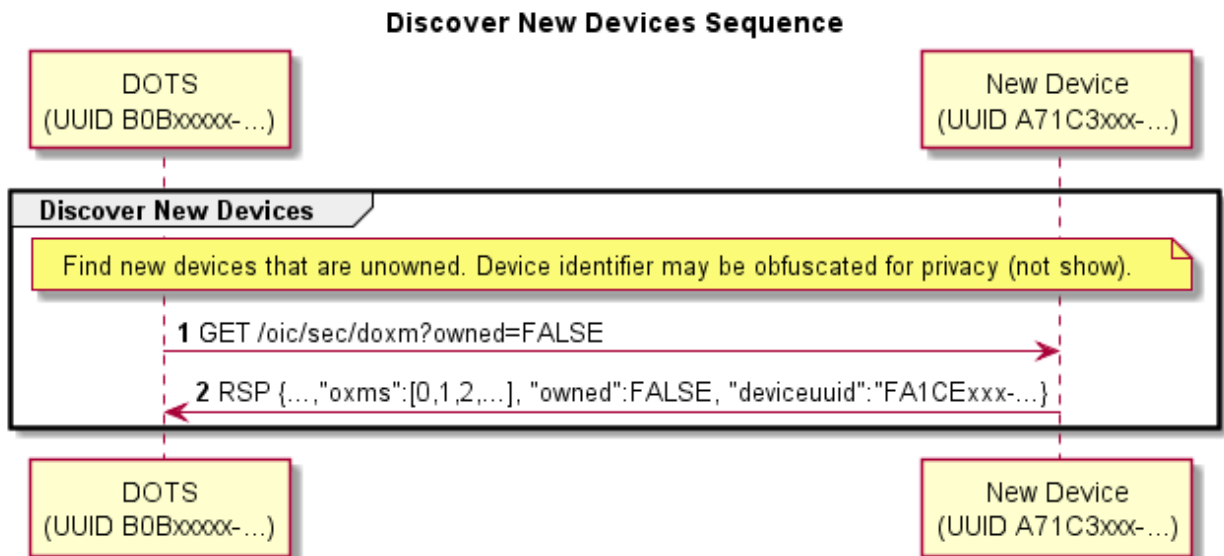
1275 **7.3.1 OTM implementation requirements**

1276 This document provides specifications for several methods for ownership transfer. Implementation
 1277 of each individual ownership transfer method is considered optional. However, each device shall
 1278 implement at least one of the ownership transfer methods not including vendor specific methods.

1279 All OTMs included in this document are considered optional. Each vendor is required to choose
 1280 and implement at least one of the OTMs specified in this document. The OCF, does however,
 1281 anticipate vendor-specific approaches will exist. Should the vendor wish to have interoperability
 1282 between a vendor-specific OTM and OBTs from other vendors, the vendor must work directly with
 1283 OBT vendors to ensure interoperability. Notwithstanding, standardization of OTMs is the preferred
 1284 approach. In such cases, a set of guidelines is provided in 7.3.7 to help vendors in designing
 1285 vendor-specific OTMs.

1286 The "/oic/sec/doxm" Resource is extensible to accommodate vendor-defined owner transfer
 1287 methods (OTM). The DOTS determines which OTM is most appropriate to onboard the new Device.
 1288 All OTMs shall represent the onboarding capabilities of the Device using the "oxms" Property of
 1289 the "/oic/sec/doxm" Resource. The DOTS queries the Device's supported credential types using
 1290 the "credtype" Property of the "/oic/sec/cred" Resource. The DOTS and CMS provision credentials
 1291 according to the credential types supported.

1292 Figure 11 depicts new Device discovery sequence.



1293

1294 **Figure 11 – Discover New Device Sequence**

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.

1297 Vendor-specific device OTMs shall adhere to the "/oic/sec/doxm" Resource Specification for OCS
1298 that results from vendor-specific device OTM. Vendor-specific OTM should include provisions for
1299 establishing trust in the new Device by the DOTS and optionally establishing trust in the OBT by
1300 the new Device.

1301 The new device may have to perform some initialization steps at the beginning of an OTM. For
1302 example, if the Random PIN Based OTM is initiated, the new device may generate a random PIN
1303 value. The DOTS updates the oxmsel property of "/oic/sec/doxm" to the value corresponding to the
1304 OTM being used, before performing other OTM steps. This update notifies the new device that
1305 ownership transfer is starting.

1306 The end state of a vendor-specific OTM shall allow the new Device to authenticate to the OBT and
1307 the OBT to authenticate to the new device.

1308 Additional provisioning steps may be performed subsequent to owner transfer success leveraging
1309 the established OTM session.

1310 **7.3.2 SharedKey Credential Calculation**

1311 The SharedKey credential is derived using a PRF that accepts the key_block value resulting from
1312 the DTLS handshake used for onboarding. The new Device shall use the following calculation to
1313 ensure interoperability across vendor products (the DOTS performs the same calculation):

1314 SharedKey = PRF(Secret, Message);

1315 Where:

- 1316 - PRF shall use TLS 1.2 PRF defined by IETF RFC 5246 clause 5.
- 1317 - Secret is the key_block resulting from the DTLS handshake
 - 1318 ▪ See IETF RFC 5246 clause 6.3
 - 1319 ▪ The length of key_block depends on cipher suite.
 - 1320 • (e.g. 96 bytes for TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
 - 1321 40 bytes for TLS_PSK_WITH_AES_128_CCM_8)
- 1322 - Message is a concatenation of the following:
 - 1323 ▪ DoxmType string for the current onboarding method (e.g. "oic.sec.doxm.jw")
 - 1324 • See clause 13.2.2 for specific DoxmTypes
 - 1325 ▪ Owner ID is a UUID identifying the device owner identifier and the device that maintains SharedKey.
 - 1326 • Use raw bytes as specified in IETF RFC 4122 clause 4.1.2
 - 1327 ▪ Device ID is new device's UUID Device ID
 - 1328 • Use raw bytes as specified in IETF RFC 4122 clause 4.1.2
- 1329 - SharedKey Length will be 32 octets.
 - 1330 ▪ If subsequent DTLS sessions use 128 bit encryption cipher suites the left most 16 octets will be used.
 - 1331 ▪ DTLS sessions using 256-bit encryption cipher suites will use all 32 octets.

1332 **7.3.3 Certificate Credential Generation**

1333 The Certificate Credential will be used by Devices for secure bidirectional communication. The
1334 certificates will be issued by a CMS or an external certificate authority (CA). This CA will be used
1335 to mutually establish the authenticity of the Device.

1336 **7.3.4 Just-Works OTM**

1337 **7.3.4.1 Just-Works OTM General**

1338 Just-works OTM creates a symmetric key credential that is a pre-shared key used to establish a
1339 secure connection through which a device should be provisioned for use within the owner's OCF
1340 Security Domain. Provisioning additional credentials and Resources is a typical step following
1341 ownership establishment. The pre-shared key is called SharedKey.

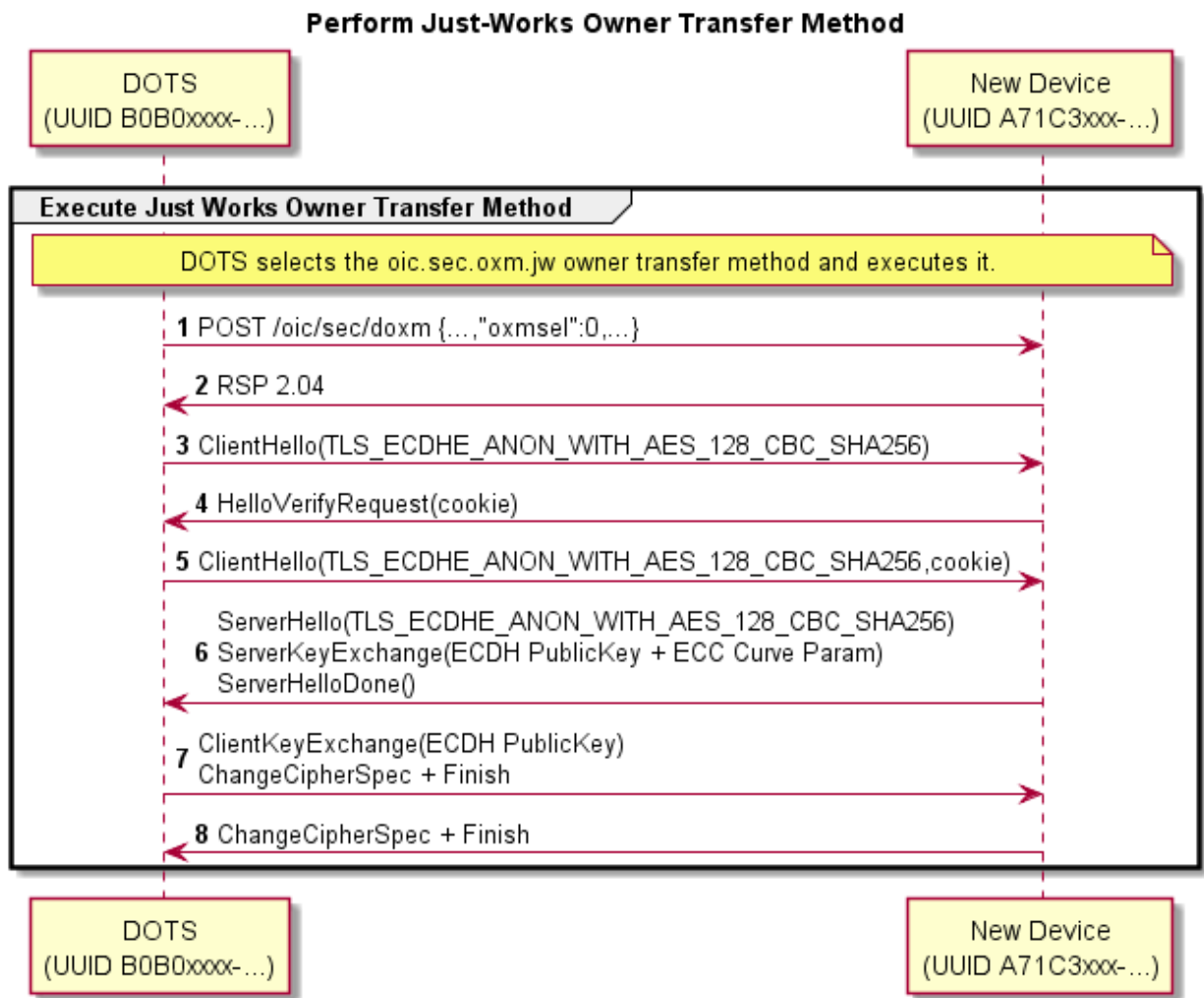
1342 The DOTS selects the Just-works OTM using the "oxmsel" Property of the "/oic/sec/doxm"
1343 Resource and establishes a DTLS session using a ciphersuite defined for the Just-works OTM.

1344 The following OCF-defined vendor-specific ciphersuites are used for the Just-works OTM.

1345 TLS_ECDH_ANON_WITH_AES_128_CBC_SHA256,
1346 TLS_ECDH_ANON_WITH_AES_256_CBC_SHA256

1347 These are not registered in IANA, the ciphersuite values are assigned from the reserved area for
1348 private use (0xFF00 ~ 0xFFFF). The assigned values are 0xFF00 and 0xFF01, respectively.

1349 Just Works OTM sequence is shown in Figure 12 and steps described in Table 2.



1350
1351
1352
1353

Figure 12 – A Just Works OTM

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. ^a
^a 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.	

1354
1355
1356
1357

1358
1359

7.3.4.2 Security Considerations

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

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

1360 The new device should use a temporal device ID prior to transitioning to an owned device while it
1361 is considered a guest device to prevent privacy sensitive tracking. The device asserts a non-
1362 temporal device ID that could differ from the temporal value during the secure session in which
1363 owner transfer exchange takes place. The DOTS verifies the asserted Device ID does not conflict
1364 with a Device ID already in use. If it is already in use the existing credentials are used to establish
1365 a secure session.

1366 An un-owned Device that also has established device credentials might be an indication of a
1367 corrupted or compromised device.

1368 **7.3.5 Random PIN Based OTM**

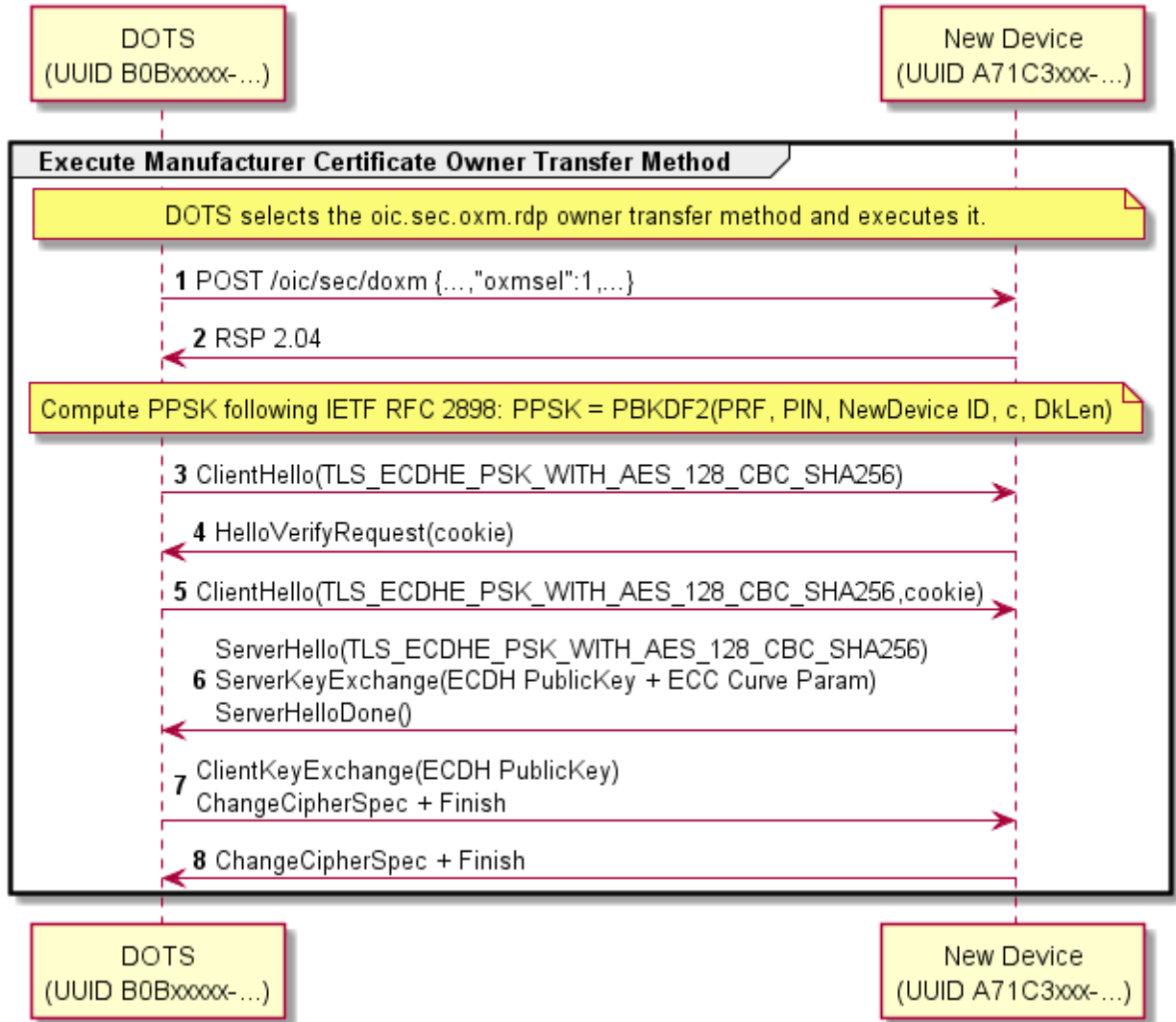
1369 **7.3.5.1 Random PIN OTM General**

1370 The Random PIN method establishes physical proximity between the new device and the OBT can
1371 prevent man-in-the-middle attacks. The Device generates a random number that is communicated
1372 to the DOTS over an Out of Band Communication Channel. The definition of an Out of Band
1373 Communication Channel is outside the scope of the definition of device OTMs. The DOTS and new
1374 Device use the PIN in a key exchange as evidence that someone authorized the transfer of
1375 ownership by having physical access to the new Device via the Out-of-Band Communication
1376 Channel.

1377 **7.3.5.2 Random PIN Owner Transfer Sequence**

1378 Random PIN-based OTM sequence is shown in Figure 13 and steps described in Table 3.

Perform Random PIN Device Owner Transfer Method



1379

1380

1381

1382

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.

1383 The random PIN-based device OTM uses a pseudo-random function (PBKDF2) defined by IETF
1384 RFC 2898 and a PIN exchanged via an Out of Band Communication Channel to generate a pre-
1385 shared key. The PIN-authenticated pre-shared key (PPSK) is supplied to TLS ciphersuites that
1386 accept a PSK.

1387 $PPSK = PBKDF2(PRF, PIN, Device\ ID, c, dkLen)$

1388 The PBKDF2 function has the following parameters:

1389 - PRF – Uses the TLS 1.2 PRF defined by IETF RFC 5246.

1390 - PIN – obtained via Out of Band Communication Channel.

1391 - Device ID – UUID of the new device.

1392 Use raw bytes as specified in IETF RFC 4122 clause 4.1.2

1393 - c – Iteration count initialized to 1000

1394 - dkLen – Desired length of the derived PSK in octets.

1395 **7.3.5.3 Security Considerations**

1396 Security of the Random PIN mechanism depends on the entropy of the PIN. Using a PIN with
1397 insufficient entropy may allow a man-in-the-middle attack to recover any long-term credentials
1398 provisioned as a part of onboarding. In particular, learning the provisioned symmetric key
1399 credentials allows an attacker to masquerade as the onboarded device.

1400 It is recommended that the entropy of the PIN be enough to withstand an online brute-force attack,
1401 40 bits or more. For example, a 12-digit numeric PIN, or an 8-character alphanumeric (0-9a-z), or
1402 a 7-character case-sensitive alphanumeric PIN (0-9a-zA-Z). A man-in-the-middle attack (MITM) is
1403 when the attacker is active on the network and can intercept and modify messages between the
1404 DOTS and device. In the MITM attack, the attacker must recover the PIN from the key exchange
1405 messages in "real time", i.e., before the peer's time out and abort the connection attempt. Having
1406 recovered the PIN, he can complete the authentication step of key exchange. The guidance given
1407 here calls for a minimum of 40 bits of entropy, however, the assurance this provides depends on
1408 the resources available to the attacker. Given the parallelizable nature of a brute force guessing
1409 attack, the attack enjoys a linear speedup as more cores/threads are added. A more conservative
1410 amount of entropy would be 64 bits. Since the Random PIN OTM requires using a DTLS ciphersuite
1411 that includes an ECDHE key exchange, the security of the Random PIN OTM is always at least
1412 equivalent to the security of the JustWorks OTM.

1413 The Random PIN OTM also has an option to use PBKDF2 to derive key material from the PIN. The
1414 rationale is to increase the cost of a brute force attack, by increasing the cost of each guess in the
1415 attack by a tuneable amount (the number of PBKDF2 iterations). In theory, this is an effective way
1416 to reduce the entropy requirement of the PIN. Unfortunately, it is difficult to quantify the reduction,
1417 since an X-fold increase in time spent by the honest peers does not directly translate to an X-fold
1418 increase in time by the attacker. This asymmetry is because the attacker may use specialized
1419 implementations and hardware not available to honest peers. For this reason, when deciding how
1420 much entropy to use for a PIN, it is recommended that implementers assume PBKDF2 provides no
1421 security, and ensure the PIN has sufficient entropy.

1422 The Random PIN device OTM security depends on an assumption that a secure Out of Band
1423 Communication Channel for communicating a randomly generated PIN from the new device to the
1424 OBT exists. If the Out of Band Communication Channel leaks some or the entire PIN to an attacker,
1425 this reduces the entropy of the PIN, and the attacks described above apply. The Out of Band
1426 Communication Channel should be chosen such that it requires proximity between the DOTS and
1427 the new device. The attacker is assumed to not have compromised the Out of Band Communication
1428 Channel. As an example Out of Band Communication Channel, the device may display a PIN to be
1429 entered into the OBT software. Another example is for the device to encode the PIN as a 2D
1430 barcode and display it for a camera on the DOTS device to capture and decode.

1431 **7.3.6 Manufacturer Certificate Based OTM**

1432 **7.3.6.1 Manufacturer Certificate Based OTM General**

1433 The manufacturer certificate-based OTM shall use a certificate embedded into the device by the
1434 manufacturer and may use a signed OBT, which determines the Trust Anchor between the device
1435 and the DOTS.

1436 Manufacturer embedded certificates do not necessarily need to chain to an OCF Root CA trust
1437 anchor.

1438 For some environments, policies or administrators, additional information about device
1439 characteristics may be sought. This list of additional attestations that OCF may or may not have
1440 tested (understanding that some attestations are incapable of testing or for which testing may be
1441 infeasible or economically unviable) can be found under the OCF Security Claims x509.v3
1442 extension described in 9.4.2.2.6.

1443 When utilizing certificate-based ownership transfer, devices shall utilize asymmetric keys with
1444 certificate data to authenticate their identities with the DOTS in the process of bringing a new
1445 device into operation on an OCF Security Domain. The onboarding process involves several
1446 discrete steps:

1447 1) Pre-on-board conditions

1448 a) The credential element of the Device's credential Resource ("/oic/sec/cred") containing the
1449 manufacturer certificate shall be identified by the "credusage" Property containing the string
1450 "oic.sec.cred.mfgcert" to indicate that the credential contains a manufacturer certificate.

1451 b) The manufacturer certificate chain shall be contained in the identified credential element's
1452 "publicdata" Property.

1453 c) The device shall contain a unique and immutable ECC asymmetric key pair.

1454 d) If the device requires authentication of the DOTS as part of ownership transfer, it is
1455 presumed that the DOTS has been registered and has obtained a certificate for its unique
1456 and immutable ECC asymmetric key pair signed by the predetermined Trust Anchor.

1457 e) User has configured the DOTS app with network access info and account info (if any).

1458 2) The DOTS authenticates the Device using ECDSA to verify the signature. Additionally, the
1459 Device may authenticate the DOTS to verify the DOTS signature.

1460 3) If authentication fails, the Device shall indicate the reason for failure and return to the Ready
1461 for OTM state. If authentication succeeds, the Device shall establish an encrypted link with the
1462 DOTS in accordance with the negotiated cipher suite.

1463 **7.3.6.2 Certificate Profiles**

1464 See 9.4.2 for details.

1465 **7.3.6.3 Certificate Owner Transfer Sequence Security Considerations**

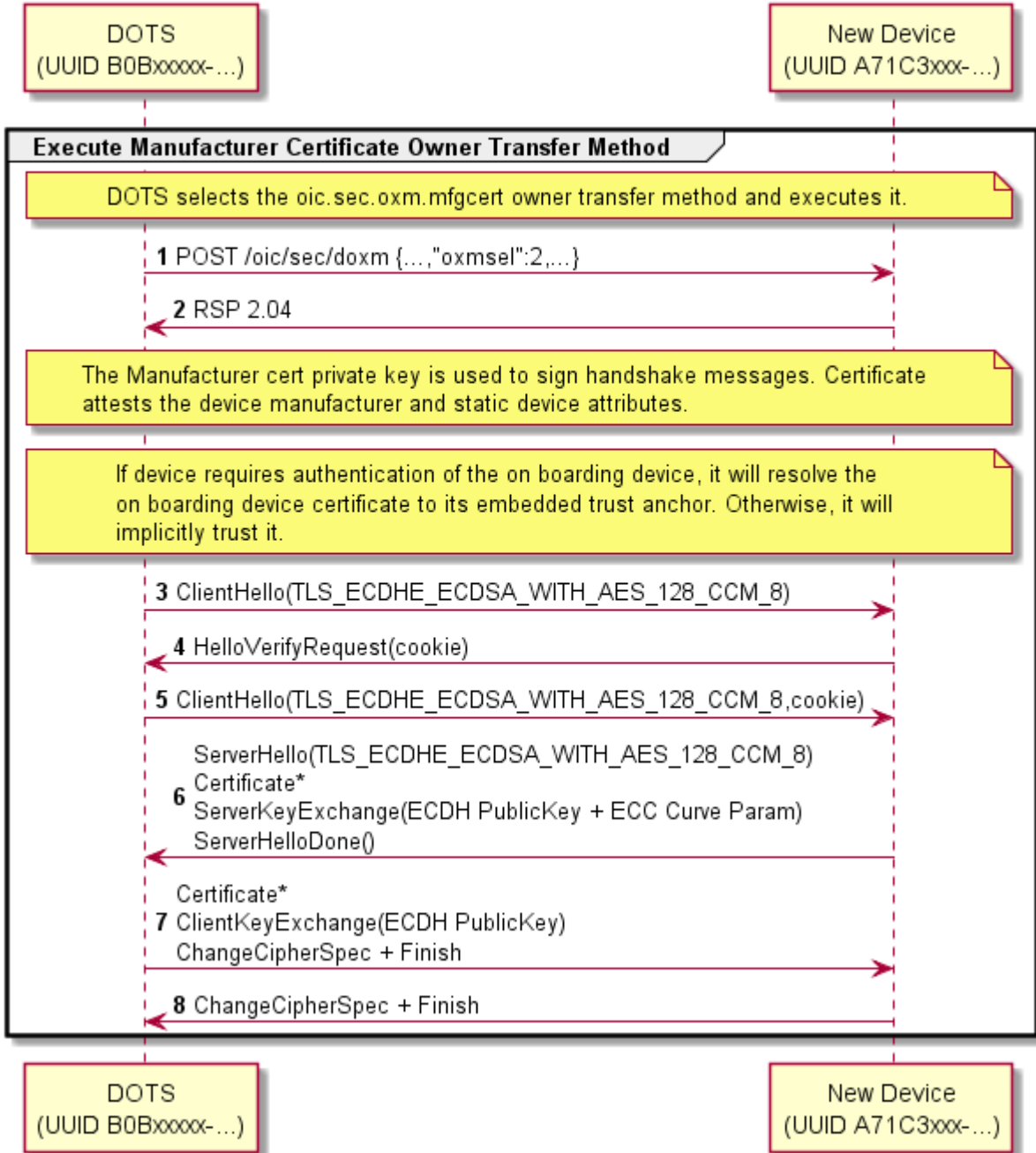
1466

1467 The OBT shall authenticate the device during onboarding. The device will not authenticate the OBT.
1468 During the DTLS handshake the server shall not send a Certificate Request.

1469 **7.3.6.4 Manufacturer Certificate Based OTM Sequence**

1470 Manufacturer Certificate Based OTM sequence is shown in Figure 14 and steps described in
1471 Table 4.

Perform Manufacturer Certificate Owner Transfer Method



1472

1473

1474

1475

Figure 14 – Manufacturer Certificate Based OTM Sequence

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.
-------	--

1476 **7.3.6.5 Security Considerations**

1477 The manufacturer certificate private key is embedded in the Platform with a sufficient degree of
 1478 assurance that the private key cannot be compromised.

1479 The Platform manufacturer issues the manufacturer certificate and attests the private key
 1480 protection mechanism.

1481 **7.3.7 Vendor Specific OTMs**

1482 **7.3.7.1 Vendor Specific OTM General**

1483 The OCF anticipates situations where a vendor will need to implement an OTM that accommodates
 1484 manufacturing or Device constraints. The Device OTM resource is extensible for this purpose.
 1485 Vendor-specific OTMs must adhere to a set of conventions that all OTMs follow.

- 1486 – The OBT must determine which credential types are supported by the Device. This is
 1487 accomplished by querying the Device's "/oic/sec/doxm" Resource to identify supported
 1488 credential types.
- 1489 – The OBT provisions the Device with OC(s).
- 1490 – The OBT supplies the Device ID and credentials for subsequent access to the OBT.
- 1491 – The OBT will supply second carrier settings sufficient for accessing the owner's OCF Security
 1492 Domain subsequent to ownership establishment.
- 1493 – The OBT may perform additional provisioning steps but must not invalidate provisioning tasks
 1494 to be performed by a security service.

1495 **7.3.7.2 Vendor-specific Owner Transfer Sequence Example**

1496 Vendor-specific OTM sequence example is shown in Figure 15 and steps described in Table 5.

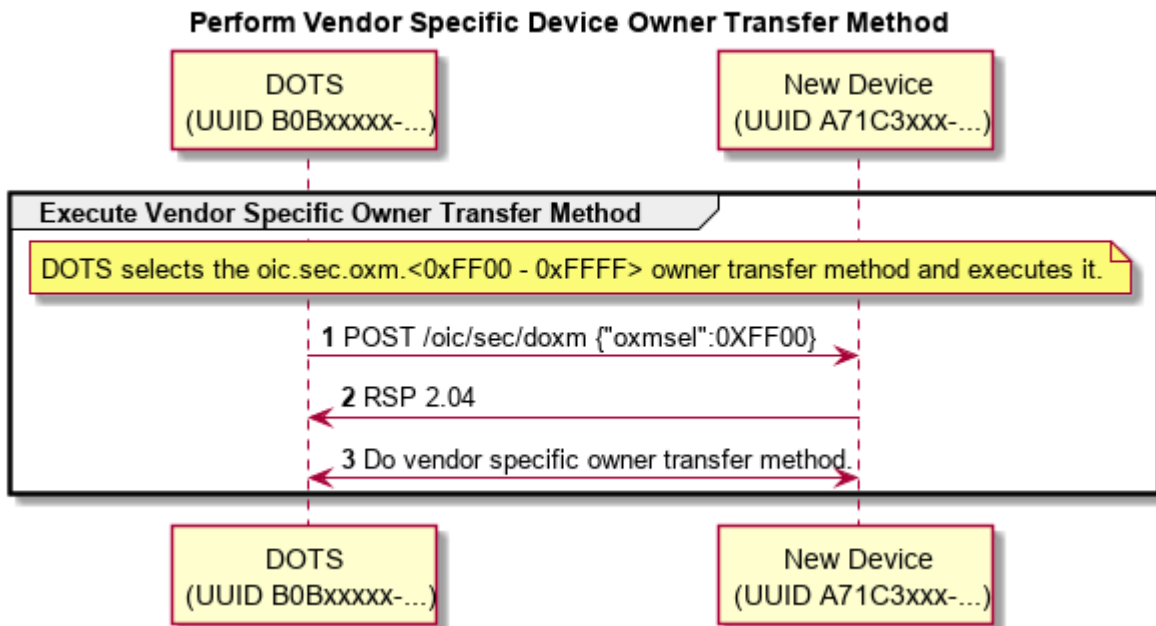


Figure 15 – Vendor-specific Owner Transfer Sequence

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

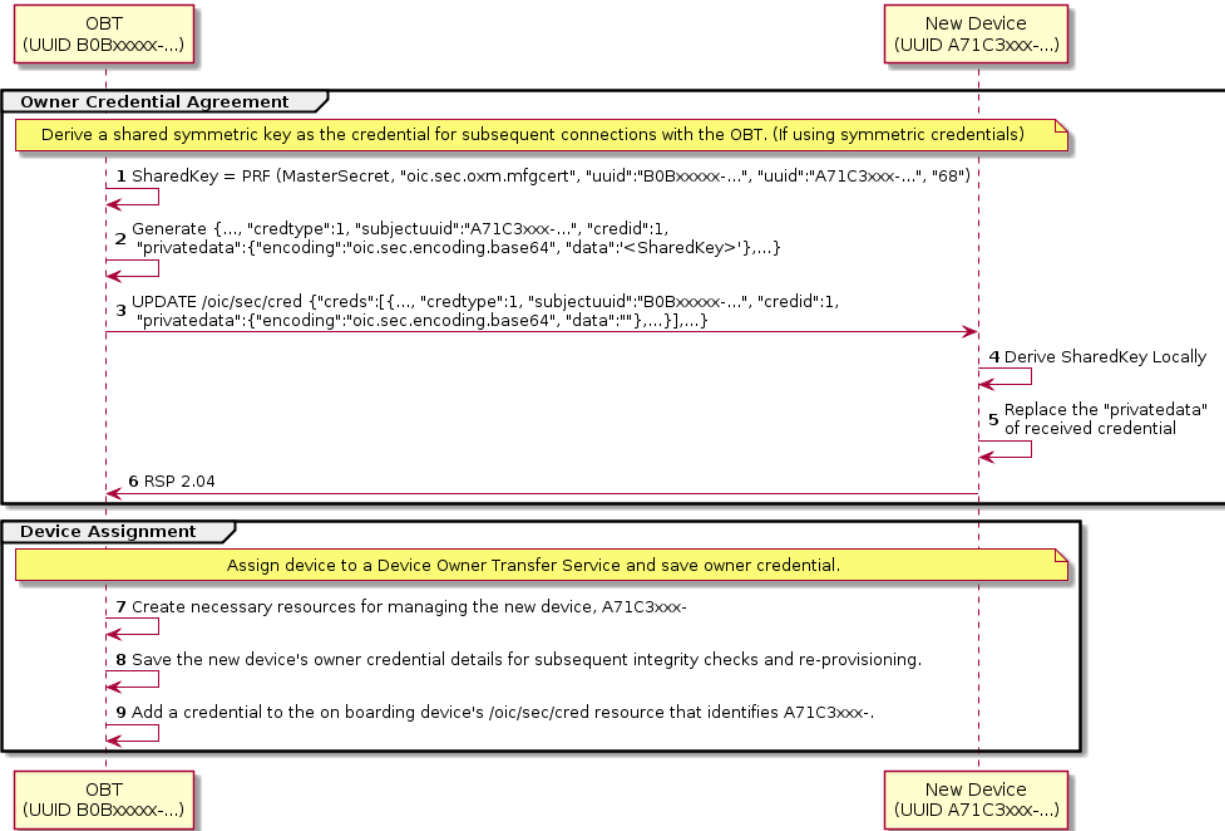
Once the OBT and the new Device have authenticated and established an encrypted connection using one of the defined OTM methods, the Owner Credential(s) can be provisioned.

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:

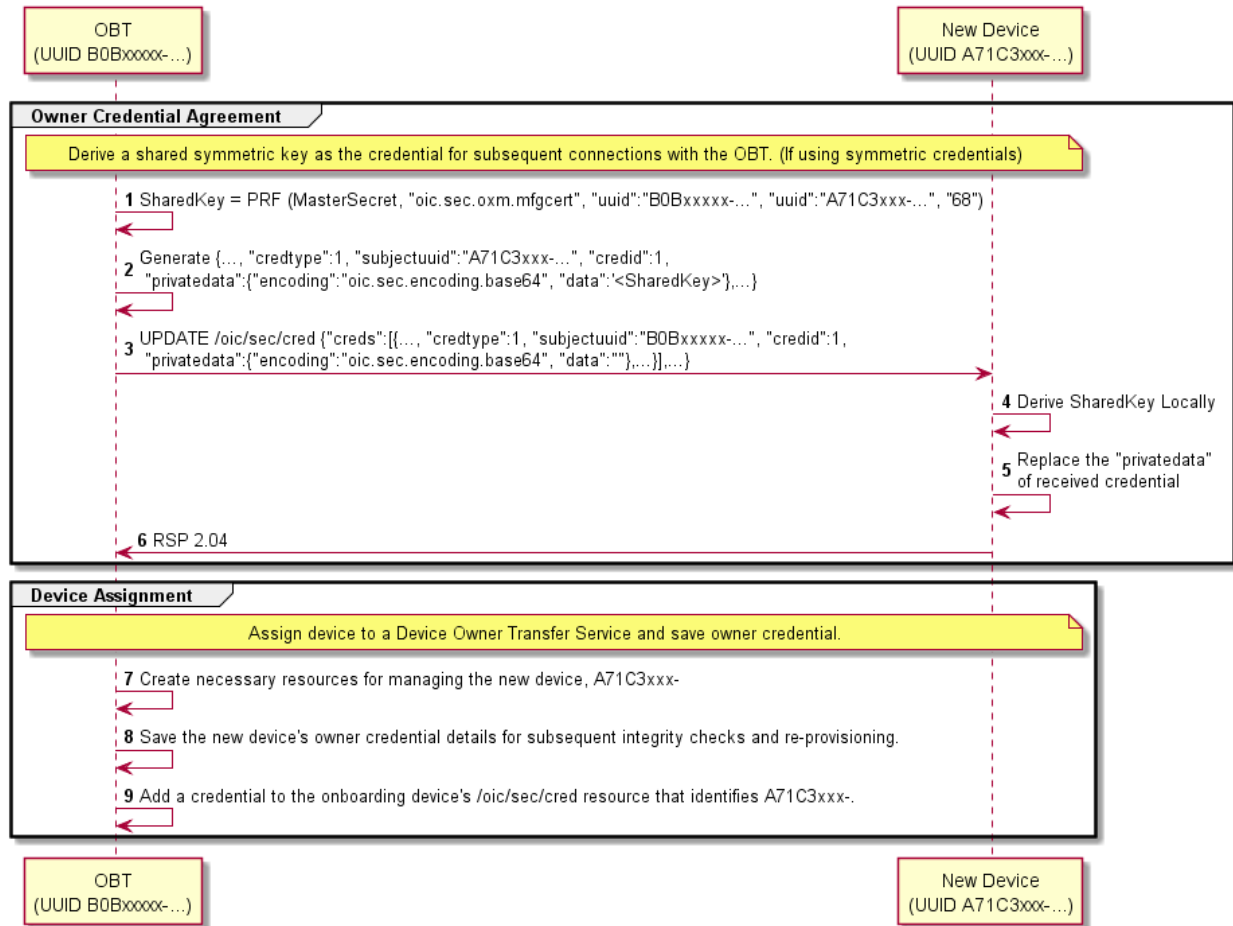
- 1) The OBT establishes the Device ID and Device Owner Id.
- 2) The OBT then establishes Device's symmetric OC - See Figure 16 and Table 6.
- 3) Configure Device services.
- 4) Configure Device for peer to peer interaction.

Symmetric Owner Credential (OC) Assignment Sequence



1514

Symmetric Owner Credential (OC) Assignment Sequence



1515

1516

1517

1518

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.

- 1519 In particular when OBT establishes symmetric owner credentials as part of OTM sequence:
- 1520 – The OBT generates a Shared Key using the SharedKey Credential Calculation method
 - 1521 described in 7.3.2.
 - 1522 – The OBT sends an empty key to the new Device's "/oic/sec/cred" Resource, identified as a
 - 1523 symmetric pair-wise key. The Subject UUID of the "/oic/sec/cred" entry shall match the Device
 - 1524 UUID of the OBT.
 - 1525 – Upon receipt of the OBT's symmetric owner credential, the new Device shall independently
 - 1526 generate the Shared Key using the SharedKey Credential Calculation method described in 7.3.2
 - 1527 and store it with the owner credential.
 - 1528 – The new Device shall use the Shared Key owner credential(s) stored via the "/oic/sec/cred"
 - 1529 Resource to authenticate the owner during subsequent connections.

1530 **7.3.9 Security considerations regarding selecting an Ownership Transfer Method -**

1531 **Moved to OCF Onboarding Tool document**

1532 This clause is intentionally left blank.

1533 **7.3.10 Security Profile Assignment**

1534 OCF Devices may have been evaluated according to an OCF Security Profile. Evaluation results

1535 could be accessed from a manufacturer's certificate, OCF web server or other public repository.

1536 The DOTS reviews evaluation results to determine which OCF Security Profiles the OCF Device is

1537 authorized to possess and configures the Device with the subset of evaluated security profiles best

1538 suited for the OCF Security Domain owner's intended segmentation strategy.

1539 The OCF Device vendor shall set a manufacturer default value for the "supportedprofiles" Property

1540 of the "/oic/sec/sp" Resource to match those approved by OCF's testing and certification process.

1541 The "currentprofile" Property of the "/oic/sec/sp" Resource shall be set to one of the values

1542 contained in the "supportedprofiles". The manufacturer default value shall be re-asserted when the

1543 Device transitions to RESET Device State.

1544 The OCF Device shall only allow the "/oic/sec/sp" Resource to be updated when the Device is in

1545 one of the following Device States: RFOTM, RFPRO, SRESET and may not allow any update as

1546 directed by a Security Profile.

1547 The DOTS may update the "supportedprofiles" Property of the "/oic/sec/sp" Resource with a subset

1548 of the OCF Security Profiles values the Device achieved as part of OCF Conformance testing. The

1549 DOTS may locate conformance results by inspecting manufacturer certificates supplied with the

1550 OCF Device by selecting the "credusage" Property of the "/oic/sec/cred" Resource having the value

1551 of "oic.sec.cred.mfgcert". The DOTS may further locate conformance results by visiting a well-

1552 known OCF web site URI corresponding to the ocfCPLAttributes extension fields (clause 9.4.2.2.7).

1553 The DOTS may select a subset of Security Profiles (from those evaluated by OCF conformance

1554 testing) based on a local policy.

1555 As part of onboarding (while the OTM session is active) the DOTS should configure ACE entries to

1556 allow DOTS access subsequent to onboarding.

1557 The DOTS should update the "currentprofile" Property of the "/oic/sec/sp" Resource with the value

1558 that most correctly depicts the OCF Security Domain owner's intended Device deployment strategy.

1559 The CMS may issue role credentials using the Security Profile value (e.g. the "sp-blue-v0 OID") to
1560 indicate the OCF Security Domain owner's intention to segment the OCF Security Domain
1561 according to a Security Profile. The CMS retrieves the supportedprofiles Property of the
1562 "/oic/sec/sp" Resource to select role names corroborated with the Device's supported Security
1563 Profiles when issuing role credentials.

1564 If the CMS issues role credentials based on a Security Profile, the AMS supplies access control
1565 entries that include the role designation(s).

1566 **7.4 Provisioning**

1567 **7.4.1 Provisioning Flows**

1568 **7.4.1.1 Provisioning Flows General**

1569 As part of onboarding a new Device a secure channel is formed between the new Device and the
1570 OBT. Subsequent to the Device ownership status being changed to "owned", there is an opportunity
1571 to begin provisioning. The OBT provisions the support services that should be subsequently used
1572 to complete Device provisioning and on-going Device management.

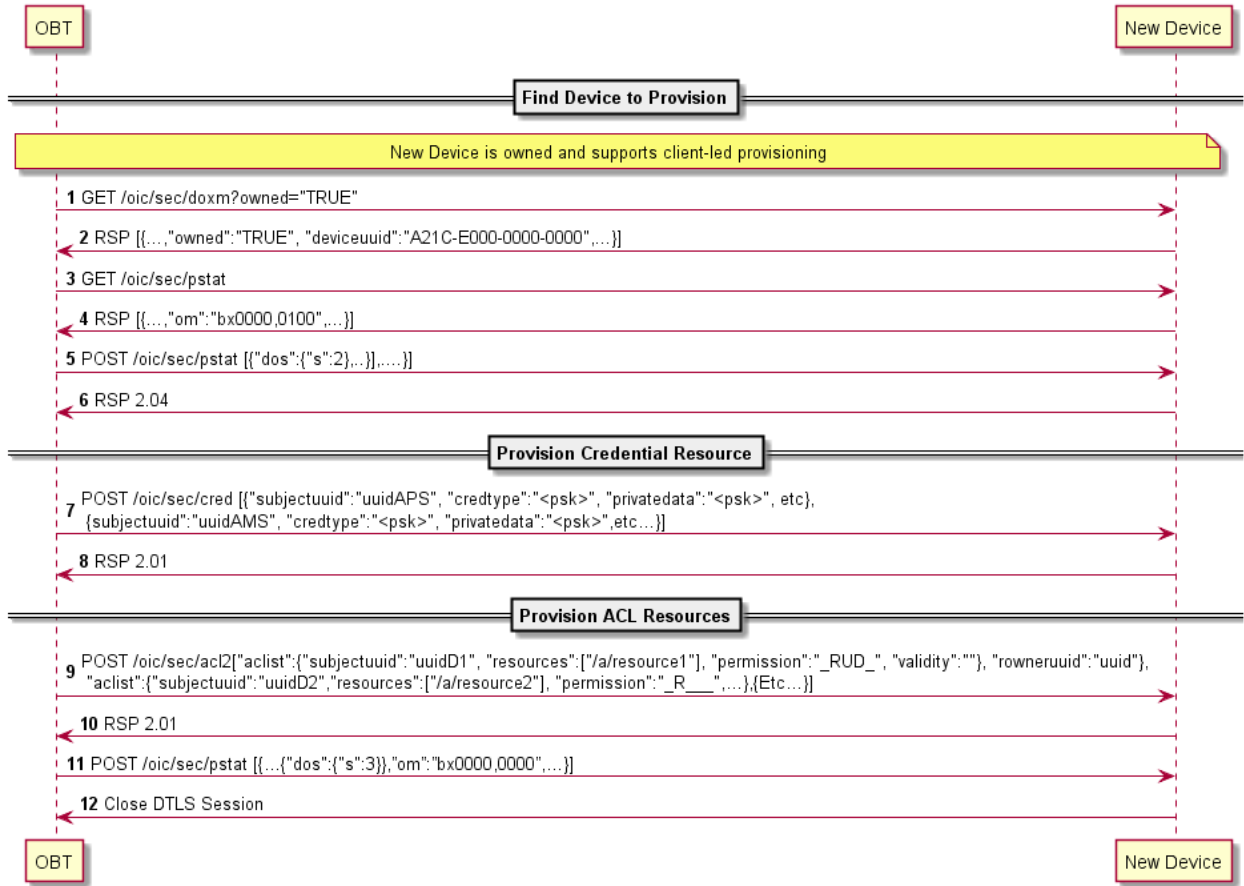
1573 The Device employs a Client-directed provisioning strategy. The "/oic/sec/pstat" Resource
1574 identifies the provisioning strategy and current provisioning status. The provisioning service should
1575 determine which provisioning strategy is most appropriate for the OCF Security Domain. See 13.8
1576 for additional detail.

1577 **7.4.1.2 Client-directed Provisioning**

1578 Client-directed provisioning relies on a provisioning service that identifies Servers in need of
1579 provisioning then performs all necessary provisioning duties.

1580 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**



1581

1582

1583

1584

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.

1585 **7.4.1.3 Server-directed Provisioning [DEPRECATED]**

1586 This clause is intentionally left blank.

1587 **7.4.1.4 Server-directed Provisioning Involving Multiple Support Services**
1588 **[DEPRECATED]**

1589 This clause is intentionally left blank.

1590 **7.5 Device Provisioning for OCF Cloud – moved to OCF Cloud Security document**

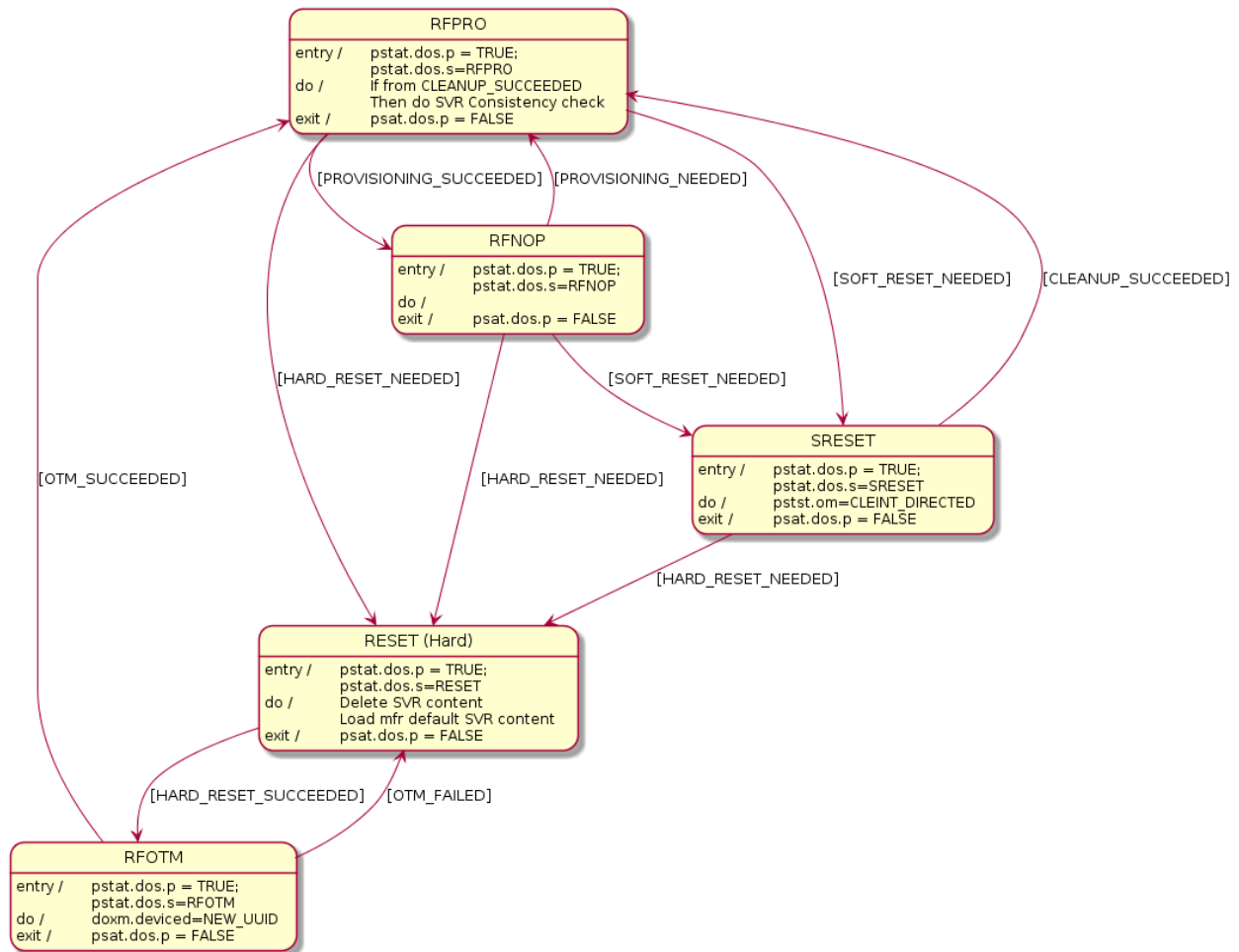
1591 This clause is intentionally left blank.

1592 **8 Device Onboarding State Definitions**

1593 **8.1 Device Onboarding General**

1594 As explained in 5.3, the process of onboarding completes after the ownership of the Device has
1595 been transferred and the Device has been provisioned with relevant configuration/services as
1596 explained in 5.4. The Figure 18 shows the various states a Device can be in during the Device
1597 lifecycle. Device shall reject any requests to perform a state transition not shown on Figure 18.

1598 The "/pstat.dos.s" Property is RW by the "/oic/sec/pstat" resource owner (e.g. "doxs" service) so
1599 that the resource owner can remotely update the Device state. When the Device is in RFNOP or
1600 RFPRO, ACLs can be used to allow remote control of Device state by other Devices. When the
1601 Device state is SRESET the Device OC may be the only indication of authorization to access the
1602 Device. The Device owner may perform low-level consistency checks and re-provisioning to get
1603 the Device suitable for a transition to RFPRO.



1604
1605

Figure 18 – Device state model

1606 As shown in the diagram, at the conclusion of the provisioning step, the Device comes in the "Ready
1607 for Normal Operation" state where it has all it needs in order to start interoperating with other
1608 Devices. Clause 8.5 specifies the minimum mandatory configuration that a Device shall hold in
1609 order to be considered as "Ready for Normal Operation".

1610 In the event of power loss or Device failure, the Device should remain in the same state that it was
1611 in prior to the power loss / failure

1612 If a Device or resource owner OBSERVes "/pstat.dos.s", then transitions to SRESET will give early
1613 warning notification of Devices that may require SVR consistency checking.

1614 In order for onboarding to function, the Device shall have the following Resources installed:

- 1615 1) "/oic/sec/doxm" Resource
- 1616 2) "/oic/sec/pstat" Resource
- 1617 3) "/oic/sec/cred" Resource

1618 The values contained in these Resources are specified in the state definitions in 8.2, 8.3, 8.4, 8.5
1619 and 8.6.

1620 **8.2 Device Onboarding-Reset State Definition**

1621 The /pstat.dos.s = RESET state is defined as a "hard" reset to manufacturer defaults. Hard reset
1622 also defines a state where the Device asset is ready to be transferred to another party.

1623 The Platform manufacturer should provide a physical mechanism (e.g. button) that forces Platform
1624 reset. All Devices hosted on the same Platform transition their Device states to RESET when the
1625 Platform reset is asserted.

1626 The following Resources and their specific properties shall have the value as specified:

- 1627 – The "owned" Property of the "/oic/sec/doxm" Resource shall transition to FALSE.
- 1628 – The "devowneruuid" Property of the "/oic/sec/doxm" Resource shall be nil UUID.
- 1629 – The "deviceuuid" Property of the "/oic/sec/doxm" Resource shall be set to the manufacturer
1630 default value.
- 1631 – The "sct" Property of the "/oic/sec/doxm" Resource shall be reset to the manufacturer's default
1632 value.
- 1633 – The "oxmsel" Property of the "/oic/sec/doxm" Resource shall be reset to the manufacturer's
1634 default value.
- 1635 – The "isop" Property of the "/oic/sec/pstat" Resource shall be FALSE.
- 1636 – The "dos" Property of the "/oic/sec/pstat" Resource shall be updated: dos.s shall equal "RESET"
1637 state and dos.p shall equal "FALSE".
- 1638 – The "om" (operational modes) Property of the "/oic/sec/pstat" Resource shall be set to the
1639 manufacturer default value.
- 1640 – The "sm" (supported operational modes) Property of the "/oic/sec/pstat" Resource shall be set
1641 to the manufacturer default value.
- 1642 – The "rowneruuid" Property of "/oic/sec/pstat", "/oic/sec/doxm", "/oic/sec/acl2", and
1643 "/oic/sec/cred" Resources shall be nil UUID.
- 1644 – The "supportedprofiles" Property of the "/oic/sec/sp" Resource shall be set to the manufacturer
1645 default value.
- 1646 – The "currentprofile" Property of the "/oic/sec/sp" Resource shall be set to the manufacturer
1647 default value.

1648 **8.3 Device Ready-for-OTM State Definition**

1649 The following Resources and their specific properties shall have the value as specified when the
1650 Device enters ready for ownership transfer:

- 1651 – The "owned" Property of the "/oic/sec/doxm" Resource shall be FALSE and will transition to
1652 TRUE.
- 1653 – The "devowneruuid" Property of the "/oic/sec/doxm" Resource shall be nil UUID.
- 1654 – The "deviceuuid" Property of the "/oic/sec/doxm" Resource shall be set to the manufacturer
1655 default value.
- 1656 – The "isop" Property of the "/oic/sec/pstat" Resource shall be FALSE.
- 1657 – The "dos" of the "/oic/sec/pstat" Resource shall be updated: "dos.s" shall equal "RFOTM" state
1658 and "dos.p" shall equal "FALSE".
- 1659 – The "/oic/sec/cred" Resource shall contain credential(s) if required by the selected OTM

1660 **8.4 Device Ready-for-Provisioning State Definition**

1661 The following Resources and their specific properties shall have the value as specified when the
1662 Device enters ready for provisioning:

- 1663 – The "owned" Property of the "/oic/sec/doxm" Resource shall be TRUE.
- 1664 – The "devowneruuid" Property of the "/oic/sec/doxm" Resource shall not be nil UUID.
- 1665 – The "deviceuuid" Property of the "/oic/sec/doxm" Resource shall not be nil UUID and shall be
1666 set to the value that was determined during RFOTM processing.
- 1667 – The "oxmsel" Property of the "/oic/sec/doxm" Resource shall have the value of the actual OTM
1668 used during ownership transfer.
- 1669 – The "isop" Property of the "/oic/sec/pstat" Resource shall be FALSE.
- 1670 – The "dos" of the "/oic/sec/pstat" Resource shall be updated: "dos.s" shall equal "RFPRO" state
1671 and "dos.p" shall equal FALSE.
- 1672 – The "rowneruuid" Property of every installed Resource shall be set to a valid Resource owner
1673 (i.e. an entity that is authorized to instantiate or update the given Resource). Failure to set a
1674 "rowneruuid" may result in an orphan Resource.
- 1675 – The "/oic/sec/cred" Resource shall contain credentials for each entity referenced by
1676 "rowneruuid" and "devowneruuid" Properties.
- 1677 – All requests to the "/oic/sec/roles" Resource received over a mutually-authenticated connection
1678 established using an identity certificate shall be granted, regardless of the configuration of the
1679 ACEs in the "/oic/sec/acl2" Resource, subject to the conditions in clause 10.4.2.

1680 **8.5 Device Ready-for-Normal-Operation State Definition**

1681 The following Resources and their specific properties shall have the value as specified when the
1682 Device enters ready for normal operation:

- 1683 – The "owned" Property of the "/oic/sec/doxm" Resource shall be TRUE.
- 1684 – The "devowneruuid" Property of the "/oic/sec/doxm" Resource shall not be nil UUID.
- 1685 – The "deviceuuid" Property of the "/oic/sec/doxm" Resource shall not be nil UUID and shall be
1686 set to the ID that was configured during OTM. Also the value of the "di" Property in "/oic/d" shall
1687 be the same as the deviceuuid.
- 1688 – The "oxmsel" Property of the "/oic/sec/doxm" Resource shall have the value of the actual OTM
1689 used during ownership transfer.
- 1690 – The "isop" Property of the "/oic/sec/pstat" Resource shall be set to TRUE by the Server once
1691 transition to RFNOP is otherwise complete.
- 1692 – The "dos" of the "/oic/sec/pstat" Resource shall be updated: "dos.s" shall equal "RFNOP" state
1693 and "dos.p" shall equal FALSE.
- 1694 – The "rowneruuid" Property of every installed Resource shall be set to a valid resource owner
1695 (i.e. an entity that is authorized to instantiate or update the given Resource). Failure to set a
1696 "rowneruuid" results in an orphan Resource.
- 1697 – The "/oic/sec/cred" Resource shall contain credentials for each service referenced by
1698 "rowneruuid" and "devowneruuid" Properties.
- 1699 – All requests to the "/oic/sec/roles" Resource received over a mutually-authenticated connection
1700 established using an identity certificate shall be granted, regardless of the configuration of the
1701 ACEs in the "/oic/sec/acl2" Resource, subject to the conditions in clause 10.4.2.

1702 **8.6 Device Soft Reset State Definition**

1703 The soft reset state is defined (e.g. "/pstat.dos.s" = SRESET) where entrance into this state means
1704 the Device is not operational but remains owned by the current owner. The Device may exit
1705 SRESET by authenticating to a DOTS (e.g. "rt" = "oic.r.doxs") using the OC provided during original
1706 onboarding (but should not require use of an OTM /doxm.oxms).

1707 If the DOTS credential cannot be found or is determined to be corrupted, the Device state
1708 transitions to RESET. The Device should remain in SRESET if the DOTS credential fails to validate
1709 the DOTS. This mitigates denial-of-service attacks that may be attempted by non-DOTS Devices.

1710 When in SRESET, the following Resources and their specific Properties shall have the values as
1711 specified.

- 1712 – The "owned" Property of the "/oic/sec/doxm" Resource shall be TRUE.
- 1713 – The "devowneruuid" Property of the "/oic/sec/doxm" Resource shall remain non-null.
- 1714 – The "deviceuuid" Property of the "/oic/sec/doxm" Resource shall remain non-null.
- 1715 – The "sct" Property of the "/oic/sec/doxm" Resource shall retain its value.
- 1716 – The "oxmsel" Property of the "/oic/sec/doxm" Resource shall retain its value.
- 1717 – The "isop" Property of the "/oic/sec/pstat" Resource shall be FALSE.
- 1718 – The "/oic/sec/pstat.dos.s" Property shall be SRESET.
- 1719 – The "om" (operational modes) Property of the "/oic/sec/pstat" Resource shall be "client-directed
1720 mode".
- 1721 – The "sm" (supported operational modes) Property of "/oic/sec/pstat" Resource may be updated
1722 by the Device owner (aka DOTS).
- 1723 – The "rowneruuid" Property of "/oic/sec/pstat", "/oic/sec/doxm", "/oic/sec/acl2", and
1724 "/oic/sec/cred" Resources may be reset by the Device owner (aka DOTS) and re-provisioned.
- 1725 – All requests to the "/oic/sec/roles" Resource received over a mutually-authenticated connection
1726 established using an identity certificate shall be granted, regardless of the configuration of the
1727 ACEs in the "/oic/sec/acl2" Resource, subject to the conditions in clause 10.4.2.

1728

1729 **9 Security Credential Management**

1730 **9.1 Preamble**

1731 This clause provides an overview of the credential types in OCF, along with details of credential
1732 use, provisioning and ongoing management.

1733 **9.2 Credential Lifecycle**

1734 **9.2.1 Credential Lifecycle General**

1735 OCF credential lifecycle has the following phases: (1) creation, (2) deletion, (3) refresh and (4)
1736 revocation.

1737 **9.2.2 Creation**

1738 The CMS can provision credentials to the credential Resource onto the Device. The Device shall
1739 verify the CMS is authorized by matching the rowneruuid Property of the "/oic/sec/cred" Resource
1740 to the DeviceID of the credential the CMS used to establish the secure connection.

1741 Credential Resources created using a CMS may involve specialized credential issuance protocols
1742 and messages. These may involve the use of public key infrastructure (PKI) such as a certificate
1743 authority (CA), symmetric key management such as a key distribution centre (KDC) or as part of a
1744 provisioning action by a DOTS, CMS or AMS.

1745 **9.2.3 Deletion**

1746 The CMS can delete credentials from the credential Resource. The Device (e.g. the Device where
1747 the credential Resource is hosted) should delete credential Resources that have expired.

1748 An expired credential Resource may be deleted to manage memory and storage space.

1749 Deletion in OCF key management is equivalent to credential suspension.

1750 **9.2.4 Refresh**

1751 Credential refresh may be performed before it expires. The CMS performs credential refresh.

1752 The "/oic/sec/cred" Resource supports expiry using the Period Property. Credential refresh may be
1753 applied when a credential is about to expire or is about to exceed a maximum threshold for bytes
1754 encrypted.

1755 A credential refresh method specifies the options available when performing key refresh. The
1756 Period Property informs when the credential should expire. The Device may proactively obtain a
1757 new credential using a credential refresh method using current unexpired credentials to refresh the
1758 existing credential. If the Device does not have an internal time source, the current time should be
1759 obtained from a CMS at regular intervals.

1760 If the onboarding established credentials are allowed to expire the DOTS shall re-onboard the
1761 Device to re-apply device owner transfer steps.

1762 All Devices shall support at least one credential refresh method.

1763 **9.2.5 Revocation**

1764 Credentials issued by a CMS may be equipped with revocation capabilities. In situations where the
1765 revocation method involves provisioning of a revocation object that identifies a credential that is to
1766 be revoked prior to its normal expiration period, a credential Resource is created containing the
1767 revocation information that supersedes the originally issued credential. The revocation object
1768 expiration should match that of the revoked credential so that the revocation object is cleaned up
1769 upon expiry.

1770 It is conceptually reasonable to consider revocation applying to a credential or to a Device. Device
1771 revocation asserts all credentials associated with the revoked Device should be considered for
1772 revocation. Device revocation is necessary when a Device is lost, stolen or compromised. Deletion
1773 of credentials on a revoked Device might not be possible or reliable.

1774 **9.3 Credential Types**

1775 **9.3.1 Preamble**

1776 The "/oic/sec/cred" Resource maintains a credential type Property that supports several
1777 cryptographic keys and other information used for authentication and data protection. The
1778 credential types supported include pair-wise symmetric keys, group symmetric keys, asymmetric
1779 authentication keys, certificates (i.e. signed asymmetric keys) and shared-secrets (i.e.
1780 PIN/password).

1781 **9.3.2 Pair-wise Symmetric Key Credentials**

1782 The CMS shall provision exactly one other pair-wise symmetric credential to a peer Device. The
1783 CMS should not store pair-wise symmetric keys it provisions to managed Devices.

1784 Pair-wise keys could be established through ad-hoc key agreement protocols.

1785 The "PrivateData" Property in the "/oic/sec/cred" Resource contains the symmetric key.

1786 The "PublicData" Property may contain a token encrypted to the peer Device containing the pair-
1787 wise key.

1788 The "OptionalData" Property may contain revocation status.

1789 The Device implementer should apply hardened key storage techniques that ensure the
1790 "PrivateData" remains private.

1791 The Device implementer should apply appropriate integrity, confidentiality and access protection
1792 of the "/oic/sec/cred", "/oic/sec/roles", "/oic/sec/csr" Resources to prevent unauthorized
1793 modifications.

1794 **9.3.3 Group Symmetric Key Credentials**

1795 Group keys are symmetric keys shared among a group of Devices (3 or more). Group keys are
1796 used for efficient sharing of data among group participants.

1797 Group keys do not provide authentication of Devices but only establish membership in a group.

1798 The CMS shall provision group symmetric key credentials to the group members. The CMS
1799 maintains the group memberships.

1800 The "PrivateData" Property in the "/oic/sec/cred" Resource contains the symmetric key.

1801 The "PublicData" Property may contain the group name.

1802 The "OptionalData" Property may contain revocation status.

1803 The Device implementer should apply hardened key storage techniques that ensure the
1804 "PrivateData" remains private.

1805 The Device implementer should apply appropriate integrity, confidentiality and access protection
1806 of the "/oic/sec/cred", "/oic/sec/roles", "/oic/sec/csr" Resources to prevent unauthorized
1807 modifications.

1808 **9.3.4 Asymmetric Authentication Key Credentials**

1809 **9.3.4.1 Asymmetric Authentication Key Credentials General**

1810 Asymmetric authentication key credentials contain either a public and private key pair or only a
1811 public key. The private key is used to sign Device authentication challenges. The public key is used
1812 to verify a device authentication challenge-response.

1813 The "PrivateData" Property in the "/oic/sec/cred" Resource contains the private key.

1814 The "PublicData" Property contains the public key.

1815 The "OptionalData" Property may contain revocation status.

1816 The Device implementer should apply hardened key storage techniques that ensure the
1817 "PrivateData" remains private.

1818 Devices should generate asymmetric authentication key pairs internally to ensure the private key
1819 is only known by the Device. See 9.3.4.2 for when it is necessary to transport private key material
1820 between Devices.

1821 The Device implementer should apply appropriate integrity, confidentiality and access protection
1822 of the "/oic/sec/cred", "/oic/sec/roles", "/oic/sec/csr" Resources to prevent unauthorized
1823 modifications.

1824 **9.3.4.2 External Creation of Asymmetric Authentication Key Credentials**

1825 Devices should employ industry-standard high-assurance techniques when allowing off-device key
1826 pair creation and provisioning. Use of such key pairs should be minimized, particularly if the key
1827 pair is immutable and cannot be changed or replaced after provisioning.

1828 When used as part of onboarding, these key pairs can be used to prove the Device possesses the
1829 manufacturer-asserted properties in a certificate to convince a DOTS or a user to accept
1830 onboarding the Device. See 7.3.3 for the OTM that uses such a certificate to authenticate the
1831 Device, and then provisions new OCF Security Domain credentials for use.

1832 **9.3.5 Asymmetric Key Encryption Key Credentials**

1833 The asymmetric key-encryption-key (KEK) credentials are used to wrap symmetric keys when
1834 distributing or storing the key.

1835 The "PrivateData" Property in the "/oic/sec/cred" Resource contains the private key.

1836 The "PublicData" Property contains the public key.

1837 The "OptionalData" Property may contain revocation status.

1838 The Device implementer should apply hardened key storage techniques that ensure the
1839 "PrivateData" remains private.

1840 The Device implementer should apply appropriate integrity, confidentiality and access protection
1841 of the "/oic/sec/cred", "/oic/sec/roles", "/oic/sec/csr" Resources to prevent unauthorized
1842 modifications.

1843 **9.3.6 Certificate Credentials**

1844 Certificate credentials are asymmetric keys that are accompanied by a certificate issued by a CMS
1845 or an external certificate authority (CA).

1846 A certificate enrolment protocol is used to obtain a certificate and establish proof-of-possession.

1847 The issued certificate is stored with the asymmetric key credential Resource.

1848 Other objects useful in managing certificate lifecycle such as certificate revocation status are
1849 associated with the credential Resource.

1850 Either an asymmetric key credential Resource or a self-signed certificate credential is used to
1851 terminate a path validation.

1852 The "PrivateData" Property in the "/oic/sec/cred" Resource contains the private key.

1853 The "PublicData" Property contains the issued certificate.

1854 The "OptionalData" Property may contain revocation status.

1855 The Device implementer should apply hardened key storage techniques that ensure the
1856 PrivateData remains private.

1857 The Device implementer should apply appropriate integrity, confidentiality and access protection
1858 of the "/oic/sec/cred", "/oic/sec/roles", "/oic/sec/csr" Resources to prevent unauthorized
1859 modifications.

1860 **9.3.7 Password Credentials**

1861 Shared secret credentials are used to maintain a PIN or password that authorizes Device access
1862 to a foreign system or Device that doesn't support any other OCF credential types.

1863 The "PrivateData" Property in the "/oic/sec/cred" Resource contains the PIN, password and other
1864 values useful for changing and verifying the password.

1865 The "PublicData" Property may contain the user or account name if applicable.

1866 The "OptionalData" Property may contain revocation status.

1867 The Device implementer should apply hardened key storage techniques that ensure the
1868 "PrivateData" remains private.

1869 The Device implementer should apply appropriate integrity, confidentiality and access protection
1870 of the "/oic/sec/cred", "/oic/sec/roles", "/oic/sec/csr" Resources to prevent unauthorized
1871 modifications.

1872 **9.4 Certificate Based Key Management**

1873 **9.4.1 Overview**

1874 To achieve authentication and transport security during communications in OCF Security Domain,
1875 certificates containing public keys of communicating parties and private keys can be used.

1876 The certificate and private key may be issued by a local or remote certificate authority (CA).

1877 The OCF certificate format is a subset of X.509 format, only elliptic curve algorithm and PEM
1878 encoding format are allowed, most of optional fields in X.509 are not supported so that the format
1879 intends to meet the constrained Device's requirement.

1880 The CMS manages the certificate lifecycle for certificates it issues. The DOTS assigns a CMS to a
1881 Device when it is newly onboarded.

1882 **9.4.2 X.509 Digital Certificate Profiles**

1883 **9.4.2.1 Digital Certificate Profile General**

1884 An OCF certificate format is a subset of X.509 format (version 3 or above) as defined in
1885 IETF RFC 5280.

1886 This clause develops a profile to facilitate the use of X.509 certificates within OCF applications for
1887 those communities wishing to make use of X.509 technology. The X.509 v3 certificate format is
1888 described in detail, with additional information regarding the format and semantics of OCF specific
1889 extension(s). The supported standard certificate extensions are also listed.

1890 Certificate Format: The OCF certificate profile is derived from IETF RFC 5280. However, this
1891 document does not support the "issuerUniqueID" and "subjectUniqueID" fields which are
1892 deprecated and shall not be used in the context of OCF. If these fields are present in a certificate,
1893 compliant entities shall ignore their contents.

1894 Certificate Encoding: Conforming entities shall use the Privacy-Enhanced Mail (PEM) to encode
1895 certificates.

1896 Certificates Hierarchy and Crypto Parameters. OCF supports a three-tier hierarchy for its Public
1897 Key Infrastructure (i.e., a Root CA, an Intermediate CA, and EE certificates). OCF accredited CAs
1898 SHALL use Elliptic Curve Cryptography (ECC) keys (secp256r1 – OID:1.2.840.10045.3.1.7) and
1899 use the ecdsaWithSHA256 (OID:1.2.840.10045.4.3.2) algorithm for certificate signatures.

1900 The following clauses specify the supported standard and custom extensions for the OCF
1901 certificates profile.

1902 **9.4.2.2 Certificate Profile and Fields**

1903 **9.4.2.2.1 Root CA Certificate Profile**

1904 Table 8 describes X.509 v1 fields required for Root CA Certificates.

1905 **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)

1906 Table 9 describes X.509 v3 extensions required for Root CA Certificates.

1907

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)

1908

9.4.2.2.2 Intermediate CA Certificate Profile

1909

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

1910

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)

1911

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

1912

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

1913 **9.4.2.2.3 End-Entity Black Certificate Profile**

1914 Table 12 describes X.509 v1 fields required for End-Entity Certificates used for Black security
1915 profile.

1916 **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)

1917 Table 13 describes X.509 v3 extensions required for End-Entity Certificates.

1918 **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	<p>End-Entity certificates chaining to an OCF Root CA SHOULD contain at least one PolicyIdentifierId set to 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"> • serverAuthentication - 1.3.6.1.5.5.7.3.1 • clientAuthentication - 1.3.6.1.5.5.7.3.2 <p>Exactly ONE of the following OIDs SHALL be present:</p> <ul style="list-style-type: none"> • Identity certificate - 1.3.6.1.4.1.44924.1.6 • Role certificate - 1.3.6.1.4.1.44924.1.7 <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:</p> <p>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</p>

			defined the semantics of the role. If the OU component is absent, the certificate issuer has defined the role. The CN 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

1919 **9.4.2.2.4 OCF Compliance X.509v3 Extension**

1920 The OCF Compliance Extension defines required parameters to correctly identify the type of Device,
1921 its manufacturer, its OCF Version, and the Security Profile compliance of the device.

1922 The extension carries an "ocfVersion" field which provides the specific base version of the OCF
1923 documents the device implements. The "ocfVersion" field shall contain a sequence of three integers
1924 ("major", "minor", and "build"). For example, if an entity is certified to be compliant with OCF
1925 specifications 1.3.2, then the "major", "minor", and "build" fields of the "ocfVersion" will be set to
1926 "1", "3", and "2" respectively. The "ocfVersion" may be used by Security Profiles to denote
1927 compliance to a specified base version of the OCF documents.

1928 The "securityProfile" field shall carry the ocfSecurityProfile OID(s) (clause 14.8.3) of one or more
1929 supported Security Profiles associated with the certificate in string form (UTF-8). All Security
1930 Profiles associated with the certificate should be identified by this field.

1931 The extension shall also carry two string fields (UTF-8): "DeviceName" and "deviceManufacturer".
1932 The fields carry human-readable descriptions of the Device's name and manufacturer, respectively.

1933 The ASN.1 definition of the OCFCompliance extension (OID – 1.3.6.1.4.1.51414.1.0) is defined as
1934 follows:

1935 id-OCF OBJECT IDENTIFIER ::= { iso(1) identified-organization(3) dod(6) internet(1)

```

1936         private(4) enterprise(1) OCF(51414) }
1937
1938     id-ocfX509Extensions OBJECT IDENTIFIER ::= { id-OCF 1 }
1939
1940     id-ocfCompliance OBJECT IDENTIFIER ::= { id-ocfX509Extensions 0 }
1941
1942     ocfVersion ::= SEQUENCE {
1943         major    INTEGER,
1944             --Major version number
1945         minor    INTEGER,
1946             --Minor version number
1947         build    INTEGER,
1948             --Build/Micro version number
1949     }
1950
1951     ocfCompliance ::= SEQUENCE {
1952         version          ocfVersion,
1953             --Device/OCF version
1954         securityProfile  SEQUENCE SIZE (1..MAX) OF ocfSecurityProfileOID,
1955             --Sequence of OCF Security Profile OID strings
1956             --Clause 14.8.2 defines valid ocfSecurityProfileOIDs
1957         deviceName      UTF8String,
1958             --Name of the device
1959         deviceManufacturer UTF8String,
1960             --Human-Readable Manufacturer
1961             --of the device
1962     }

```

1963 **9.4.2.2.5 Manufacturer Usage Description (MUD) X.509v3 Extension**

1964 The goal of the Manufacturer Usage Description (MUD) extension is to provide a means for devices
1965 to signal to the network the access and network functionality they require to properly function.
1966 Access controls can be more easily achieved and deployed at scale when the MUD extension is
1967 used.

1968 The MUD X.509 v3 extension is specified in IETF RFC 8520 with the full ASN.1 definition in section
1969 11.

1970 **9.4.2.2.6 OCF Security Claims X.509v3 Extension**

1971 The OCF Security Claims Extension defines a list of OIDs representing security claims that the
1972 manufacturer/integrator is making as to the security posture of the device above those required by
1973 the OCF Compliance version or that of the OCF Security Profile being indicated by the device.

1974 The purpose of this extension is to allow for programmatic evaluation of assertions made about
1975 security to enable some platforms/policies/administrators to better understand what is being
1976 onboarded or challenged.

1977 The ASN.1 definition of the OCF Security Claims extension (OID – 1.3.6.1.4.1.51414.1.1) is defined
1978 as follows:

```

1979     id-OCF OBJECT IDENTIFIER ::= { iso(1) identified-organization(3) dod(6) internet(1)
1980         private(4) enterprise(1) OCF(51414) }
1981
1982     id-ocfX509Extensions OBJECT IDENTIFIER ::= { id-OCF 1 }
1983
1984     id-ocfSecurityClaims OBJECT IDENTIFIER ::= { id-ocfX509Extensions 1 }
1985
1986         claim-secure-boot ::= ocfSecurityClaimsOID { id-ocfSecurityClaims 0 }
1987         --Device claims that the boot process follows a procedure trusted
1988         --by the firmware and the BIOS
1989

```

1990 claim-hw-backed-cred-storage ::= ocfSecurityClaimsOID { id-ocfSecurityClaims 1 }
1991 --Device claims that credentials are stored in a specialized hardware
1992 --protection environment such as a Trusted Platform Module (TPM) or
1993 --similar mechanism.

1995 ocfSecurityClaimsOID ::= OBJECT IDENTIFIER

1996
1997 ocfSecurityClaims ::= SEQUENCE SIZE (1..MAX) of ocfSecurityClaimsOID

1998 **9.4.2.2.7 OCF Certified Product List Attributes X.509v3 Extension**

1999 The OCF Certified Product List Extension defines required parameters to utilize the OCF
2000 Compliance Management System Certified Product List (OCMS-CPL). This clause is only
2001 applicable if you plan to utilize the OCMS-CPL. The OBT may make use of these attributes to verify
2002 the compliance level of a device.

2003 The extension carries the OCF CPL Attributes: IANA Private Enterprise Number (PEN), Model and
2004 Version.

2005 The 'cpl-at-IANAPen' IANA Private Enterprise Number (PEN) provides the manufacturer's unique
2006 PEN established in the IANA PEN list located at: [https://www.iana.org/assignments/enterprise-](https://www.iana.org/assignments/enterprise-numbers)
2007 numbers. The 'cpl-at-IANAPen' field found in end-products shall be the same information as
2008 reported during OCF Certification.

2009 The 'cpl-at-model' represents an OCF-Certified product's model name. The 'cpl-at-model' field
2010 found in end-products shall be the same information as reported during OCF Certification.

2011 The 'cpl-at-version' represents an OCF-Certified product's version. The 'cpl-at-version' field found
2012 in end-products shall be the same information as reported during OCF Certification.

2013 The ASN.1 definition of the OCF CPL Attributes extension (OID – 1.3.6.1.4.1.51414.1.2) is defined
2014 as follows:

2015 id-OCF OBJECT IDENTIFIER ::= { iso(1) identified-organization(3) dod(6) internet(1)
2016 private(4) enterprise(1) OCF(51414) }

2017 id-ocfX509Extensions OBJECT IDENTIFIER ::= { id-OCF 1 }

2018
2019 id-ocfCPLAttributes OBJECT IDENTIFIER ::= { id-ocfX509Extensions 2 }

2020
2021 cpl-at-IANAPen ::= OBJECT IDENTIFIER { id-ocfCPLAttributes 0 }

2022 cpl-at-model ::= OBJECT IDENTIFIER { id-ocfCPLAttributes 1 }

2023 cpl-at-version ::= OBJECT IDENTIFIER { id-ocfCPLAttributes 2 }

2024
2025
2026

2027 ocfCPLAttributes ::= SEQUENCE {
2028 cpl-at-IANAPen UTF8String,
2029 --Manufacturer's registered IANA Private Enterprise Number
2030 cpl-at-model UTF8String,
2031 --Device OCF Security Profile
2032 cpl-at-version UTF8String
2033 --Name of the device
2034 }

2035 **9.4.2.3 Supported Certificate Extensions**

2036 As these certificate extensions are a standard part of IETF RFC 5280, this document includes the
2037 clause number from that RFC to include it by reference. Each extension is summarized here, and
2038 any modifications to the RFC definition are listed. Devices MUST implement and understand the
2039 extensions listed here; other extensions from the RFC are not included in this document and

2040 therefore are not required. 10.4 describes what Devices must implement when validating certificate
2041 chains, including processing of extensions, and actions to take when certain extensions are absent.

2042 – Authority Key Identifier (4.2.1.1)

2043 The Authority Key Identifier (AKI) extension provides a means of identifying the public key
2044 corresponding to the private key used to sign a certificate. This document makes the following
2045 modifications to the referenced definition of this extension:

2046 The "authorityCertIssuer" or "authorityCertSerialNumber" fields of the "AuthorityKeyIdentifier"
2047 sequence are not permitted; only "keyIdentifier" is allowed. This results in the following
2048 grammar definition:

```
2049 id-ce-authorityKeyIdentifier OBJECT IDENTIFIER ::= { id-ce 35 }
```

```
2050 AuthorityKeyIdentifier ::= SEQUENCE {  
2051     keyIdentifier          [0] KeyIdentifier          }
```

```
2052 KeyIdentifier ::= OCTET STRING  
2053  
2054
```

2055 – Subject Key Identifier (4.2.1.2)

2056 The Subject Key Identifier (SKI) extension provides a means of identifying certificates that
2057 contain a particular public key.

2058 This document makes the following modification to the referenced definition of this extension:

2059 Subject Key Identifiers SHOULD be derived from the public key contained in the certificate's
2060 "SubjectPublicKeyInfo" field or a method that generates unique values. This document
2061 RECOMMENDS the 256-bit SHA-2 hash of the value of the BIT STRING "subjectPublicKey"
2062 (excluding the tag, length, and number of unused bits). Devices verifying certificate chains must
2063 not assume any particular method of computing key identifiers, however, and must only base
2064 matching AKI's and SKI's in certification path constructions on key identifiers seen in certificates.

2065 – Subject Alternative Name

2066 If the EKU extension is present, and has the value XXXXXX, indicating that this is a role
2067 certificate, the Subject Alternative Name (subjectAltName) extension shall be present and
2068 interpreted as described below. When no EKU is present, or has another value, the
2069 "subjectAltName" extension SHOULD be absent. The "subjectAltName" extension is used to
2070 encode one or more Role ID values in role certificates, binding the roles to the subject public
2071 key. The "subjectAltName" extension is defined in IETF RFC 5280 (See 4.2.1.6):

```
2072 id-ce-subjectAltName OBJECT IDENTIFIER ::= { id-ce 17 }
```

```
2073 SubjectAltName ::= GeneralNames  
2074
```

```
2075 GeneralNames ::= SEQUENCE SIZE (1..MAX) OF GeneralName  
2076
```

```
2077 GeneralName ::= CHOICE {  
2078     otherName          [0] OtherName,  
2079     rfc5322Name       [1] IA5String,  
2080     dNSName           [2] IA5String,  
2081     x400Address       [3] ORAddress,  
2082     directoryName     [4] Name,  
2083     ediPartyName      [5] EDIPartyName,  
2084     uniformResourceIdentifier [6] IA5String,  
2085     ipAddress         [7] OCTET STRING,  
2086     registeredID      [8] OBJECT IDENTIFIER }
```

```
2087  
2088 EDIPartyName ::= SEQUENCE {  
2089     nameAssigner      [0] DirectoryString OPTIONAL,  
2090     partyName         [1] DirectoryString }
```

2092

2093 Each "GeneralName" in the "GeneralNames" SEQUENCE which encodes a role shall be a
2094 "directoryName", which is of type Name. Name is an X.501 Distinguished Name. Each Name
2095 shall contain exactly one CN (Common Name) component, and zero or one OU (Organizational
2096 Unit) components. The OU component, if present, shall specify the authority that defined the
2097 semantics of the role. If the OU component is absent, the certificate issuer has defined the role.
2098 The CN component shall encode the role ID. Other "GeneralName" types in the SEQUENCE
2099 may be present, but shall not be interpreted as roles. Therefore, if the certificate issuer includes
2100 non-role names in the "subjectAltName" extension, the extension should not be marked critical.

2101 The role, and authority need to be encoded as ASN.1 "PrintableString" type, the restricted
2102 character set [0-9a-z-A-z '()+,.-/:=?].

2103 – Key Usage (4.2.1.3)

2104 The key usage extension defines the purpose (e.g., encipherment, signature, certificate signing)
2105 of the key contained in the certificate. The usage restriction might be employed when a key that
2106 could be used for more than one operation is to be restricted.

2107 This document does not modify the referenced definition of this extension.

2108 – Basic Constraints (4.2.1.9)

2109 The basic constraints extension identifies whether the subject of the certificate is a CA and the
2110 maximum depth of valid certification paths that include this certificate. Without this extension,
2111 a certificate cannot be an issuer of other certificates.

2112 This document does not modify the referenced definition of this extension.

2113 – Extended Key Usage (4.2.1.12)

2114 Extended Key Usage describes allowed purposes for which the certified public key may can be
2115 used. When a Device receives a certificate, it determines the purpose based on the context of
2116 the interaction in which the certificate is presented, and verifies the certificate can be used for
2117 that purpose.

2118

2119 This document makes the following modifications to the referenced definition of this extension:
2120 CAs SHOULD mark this extension as critical.

2121 CAs MUST NOT issue certificates with the anyExtendedKeyUsage OID (2.5.29.37.0).

2122

2123 The list of OCF-specific purposes and the assigned OIDs to represent them are:

2124 – Identity certificate 1.3.6.1.4.1.44924.1.6
2125 – Role certificate 1.3.6.1.4.1.44924.1.7

2126 **9.4.2.4 Cipher Suite for Authentication, Confidentiality and Integrity**

2127 OCF compliant entities shall support TLS version 1.2. Compliant entities shall support
2128 TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8 cipher suite as defined in IETF RFC 7251 and may
2129 support additional ciphers as defined in the TLS v1.2 specifications.

2130 **9.4.2.5 Encoding of Certificate**

2131 See 9.4.2 for details.

2132 **9.4.3 Certificate Revocation List (CRL) Profile [Deprecated]**

2133 This clause is intentionally left blank.

2134 **9.4.4 Resource Model**

2135 Device certificates and private keys are kept in "cred" Resource.

2136 The "cred" Resource contains the certificate information pertaining to the Device. The "PublicData"
2137 Property holds the device certificate and CA certificate chain. "PrivateData" Property holds the
2138 Device private key paired to the certificate. (See 13.3 for additional detail regarding the
2139 "/oic/sec/cred" Resource).

2140 **9.4.5 Certificate Provisioning**

2141 The CMS (e.g. a hub or a smart phone) issues certificates for new Devices.

2142 The CA in the CMS retrieves a Device's public key and proof of possession of the private key,
2143 generates a Device's certificate signed by this CA certificate, and then the CMS transfers them to
2144 the Device including its CA certificate chain. Optionally, the CMS can also transfer one or more
2145 role certificates, which shall have the format described in clause 9.4.2. The "subjectPublicKey" of
2146 each role certificate shall match the "subjectPublicKey" in the Device certificate.

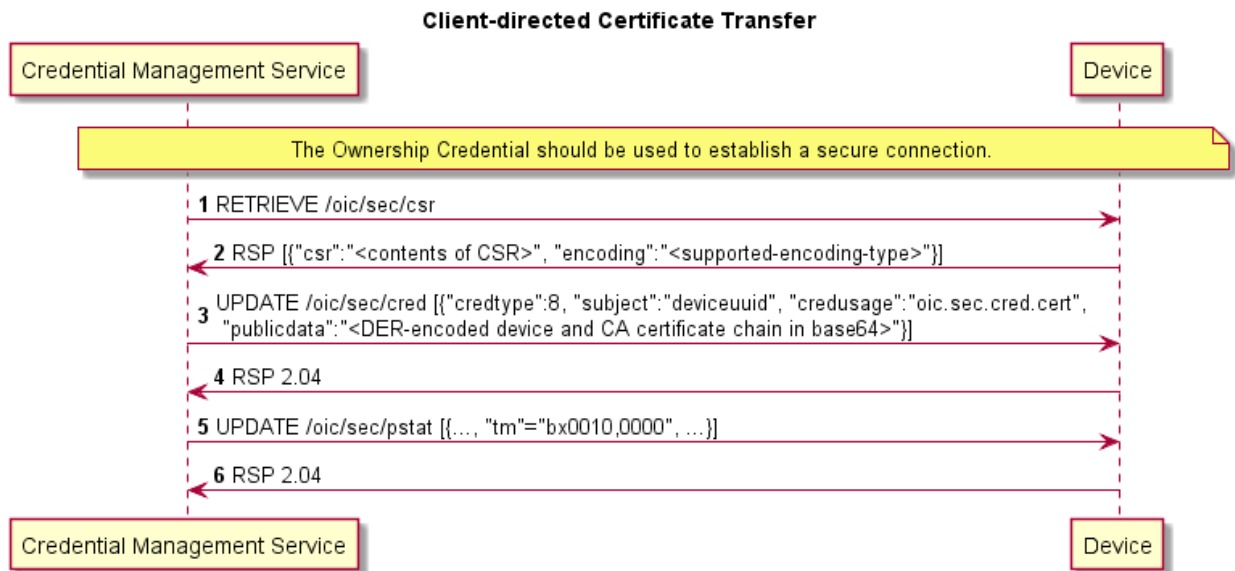
2147 In the sequence in Figure 19, the Certificate Signing Request (CSR) is defined by PKCS#10 in
2148 IETF RFC 2986, and is included here by reference.

2149 The sequence flow of a certificate transfer for a Client-directed model is described in Figure 19.

2150 1) The CMS retrieves a CSR from the Device that requests a certificate. In this CSR, the Device
2151 shall place its requested UUID into the subject and its public key in the "SubjectPublicKeyInfo".
2152 The Device determines the public key to present; this may be an already-provisioned key it has
2153 selected for use with authentication, or if none is present, it may generate a new key pair
2154 internally and provide the public part. The key pair shall be compatible with the allowed
2155 ciphersuites listed in 9.4.2.4 and 11.3.4, since the certificate will be restricted for use in OCF
2156 authentication.

2157 2) If the Device does not have a pre-provisioned key pair and is unable to generate a key pair on
2158 its own, then it is not capable of using certificates. The Device shall advertise this fact both by
2159 setting the 0x8 bit position in the sct Property of "/oic/sec/doxm" to 0, and return an error that
2160 the "/oic/sec/csr" resource does not exist.

2161 3) The CMS transfers the issued certificate and CA chain to the designated Device using the same
2162 credid, to maintain the association with the private key. The credential type ("oic.sec.cred")
2163 used to transfer certificates in Figure 19 is also used to transfer role certificates, by including
2164 multiple credentials in the POST from CMS to Device. Identity certificates shall be stored with
2165 the credusage Property set to "oic.sec.cred.cert" and role certificates shall be stored with the
2166 credusage Property set to "oic.sec.cred.rolecert".



2167

2168

Figure 19 – Client-directed Certificate Transfer

2169 **9.4.6 CRL Provisioning [Deprecated]**

2170 This clause is intentionally left blank.

2171

2172 **10 Device Authentication**

2173 **10.1 Device Authentication General**

2174 When a Client is accessing a restricted Resource on a Server, the Server shall authenticate the
2175 Client. Clients shall authenticate Servers while requesting access. Clients may also assert one or
2176 more roles that the server can use in access control decisions. Roles may be asserted when the
2177 Device authentication is done with certificates.

2178 **10.2 Device Authentication with Symmetric Key Credentials**

2179 When using symmetric keys to authenticate, the Server Device shall include the
2180 ServerKeyExchange message and set `psk_identity_hint` to the Server's Device ID. The Client shall
2181 validate that it has a credential with the Subject UUID set to the Server's Device ID, and a credential
2182 type of PSK. If it does not, the Client shall respond with an `unknown_psk_identity` error or other
2183 suitable error.

2184 If the Client finds a suitable PSK credential, it shall reply with a ClientKeyExchange message that
2185 includes a `psk_identity` set to the Client's Device ID. The Server shall verify that it has a credential
2186 with the matching Subject UUID and type. If it does not, the Server shall respond with an
2187 `unknown_psk_identity` or other suitable error code. If it does, then it shall continue with the DTLS
2188 protocol, and both Client and Server shall compute the resulting premaster secret.

2189 **10.3 Device Authentication with Raw Asymmetric Key Credentials**

2190 When using raw asymmetric keys to authenticate, the Client and the Server shall include a suitable
2191 public key from a credential that is bound to their Device. Each Device shall verify that the provided
2192 public key matches the `PublicData` field of a credential they have, and use the corresponding
2193 Subject UUID of the credential to identify the peer Device.

2194 **10.4 Device Authentication with Certificates**

2195 **10.4.1 Device Authentication with Certificates General**

2196 When using certificates to authenticate, the Client and Server shall each include their certificate
2197 chain, as stored in the appropriate credential, as part of the selected authentication cipher suite.
2198 Each Device shall validate the certificate chain presented by the peer Device. Each certificate
2199 signature shall be verified until a public key is found within the `"/oic/sec/cred"` Resource with the
2200 `"oic.sec.cred.trustca"` credusage. Credential Resource found in `"/oic/sec/cred"` is used to terminate
2201 certificate path validation. Also, the validity period and revocation status should be checked for all
2202 above certificates.

2203 A Device retrieves the Subject UUID from the Common Name component of the Subject Name
2204 property of the End-Entity certificate which has the following format: `"uuid: X"`, where X is
2205 provisioned by the CMS to match the `"deviceuuid"` Property of the `"/oic/sec/doxm"` Resource. The
2206 Device treats all requests arriving over a connection authenticated by this End-Entity certificate as
2207 having originated from the Device with this Subject UUID. The Device shall use this Subject UUID
2208 to match against the `"subjectuuid"` Property of the provisioned ACL entries to perform access
2209 control checks.

2210 Devices must follow the certificate path validation algorithm in clause 6 of IETF RFC 5280. In
2211 particular:

- 2212 – For all non-End-Entity certificates, Devices shall verify that the basic constraints extension is
2213 present, and that the `cA` boolean in the extension is `TRUE`. If either is false, the certificate chain
2214 MUST be rejected. If the `pathLenConstraint` field is present, Devices will confirm the number of
2215 certificates between this certificate and the End-Entity certificate is less than or equal to
2216 `pathLenConstraint`. In particular, if `pathLenConstraint` is zero, only an End-Entity certificate can
2217 be issued by this certificate. If the `pathLenConstraint` field is absent, there is no limit to the
2218 chain length.

- 2219 – For all non-End-Entity certificates, Devices shall verify that the key usage extension is present,
2220 and that the keyCertSign bit is asserted.
- 2221 – Devices may use the Authority Key Identifier extension to quickly locate the issuing certificate.
2222 Devices MUST NOT reject a certificate for lacking this extension, and must instead attempt
2223 validation with the public keys of possible issuer certificates whose subject name equals the
2224 issuer name of this certificate.
- 2225 – The End-Entity certificate of the chain shall be verified to contain an Extended Key Usage (EKU)
2226 suitable to the purpose for which it is being presented. An End-Entity certificate which contains
2227 no ECU extension is not valid for any purpose and must be rejected. Any certificate which
2228 contains the anyExtendedKeyUsage OID (2.5.29.37.0) must be rejected, even if other valid
2229 EKUs are also present.
- 2230 – Devices MUST verify "transitive ECU" for certificate chains. Issuer certificates (any certificate
2231 that is not an End-Entity) in the chain MUST all be valid for the purpose for which the certificate
2232 chain is being presented. An issuer certificate is valid for a purpose if it contains an ECU
2233 extension and the ECU OID for that purpose is listed in the extension, OR it does not have an
2234 ECU extension. An issuer certificate SHOULD contain an ECU extension and a complete list of
2235 EKUs for the purposes for which it is authorized to issue certificates. An issuer certificate
2236 without an ECU extension is valid for all purposes; this differs from End-Entity certificates
2237 without an ECU extension.
- 2238 The list of purposes and their associated OIDs are defined in 9.4.2.3.

2239 If the Device does not recognize an extension, it must examine the "critical" field. If the field is
2240 TRUE, the Device MUST reject the certificate. If the field is FALSE, the Device MUST treat the
2241 certificate as if the extension were absent and proceed accordingly. This applies to all certificates
2242 in a chain.

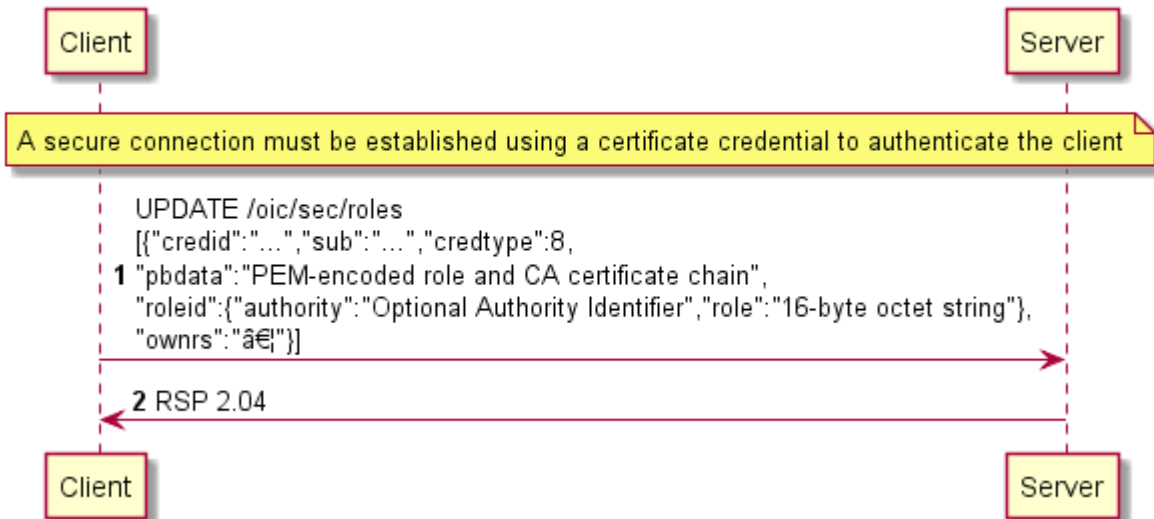
2243 NOTE Certificate revocation mechanisms are currently out of scope of this version of the document.

2244 **10.4.2 Role Assertion with Certificates**

2245 This clause describes role assertion by a client to a server using a certificate role credential. If a
2246 server does not support the certificate credential type, clients should not attempt to assert roles
2247 with certificates.

2248 Following authentication with a certificate, an OCF Client shall assert roles by updating the server's
2249 "/oic/sec/roles" Resource with all the role certificates it possesses, unless the device manufacturer
2250 provides a vendor-specific mechanism for End User to select which roles to assert. The role
2251 credentials must be certificate credentials and shall include a certificate chain. The server shall
2252 validate each certificate chain as specified in clause 10.3. Additionally, the public key in the End-
2253 Entity certificate used for Device authentication must be identical to the public key in all role (End-
2254 Entity) certificates. Also, the subject distinguished name in the End-Entity authentication and role
2255 certificates must match. The roles asserted are encoded in the subjectAltName extension in the
2256 certificate. The "subjectAltName" field can have multiple values, allowing a single certificate to
2257 encode multiple roles that apply to the client. The server shall also check that the ECU extension
2258 of the role certificate(s) contains the value 1.3.6.1.4.1.44924.1.7 (see clause 9.4.2.2) indicating the
2259 certificate may be used to assert roles. Figure 20 describes how a client Device asserts roles to a
2260 server.

Asserting Certificate Role Credentials



2261

2262

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

2263 Additional comments for Figure 20

- 2264 1) The response shall contain "204 No Content" to indicate success or 4xx to indicate an error. If
2265 the server does not support certificate credentials, it should return "501 Not Implemented"
- 2266 2) Roles asserted by the client may be kept for a duration chosen by the server. The duration shall
2267 not exceed the validity period of the role certificate.
- 2268 3) Servers should choose a nonzero duration to avoid the cost of frequent re-assertion of a role
2269 by a client. It is recommended that servers use the validity period of the certificate as a duration,
2270 effectively allowing the CMS to decide the duration.
- 2271 4) The format of the data sent in the create call shall be a list of credentials ("oic.sec.cred", see
2272 Table 19). They shall have "credtype" 8 (indicating certificates) and "PrivateData" field shall
2273 not be present. For fields that are duplicated in the "oic.sec.cred" object and the certificate, the
2274 value in the certificate shall be used for validation. For example, if the "Period" field is set in
2275 the credential, the server shall treat the validity period in the certificate as authoritative. Similar
2276 for the roleid data (authority, role).
- 2277 5) Certificates shall be encoded as in Figure 19 (PEM-encoded certificate chain).
- 2278 6) Clients may GET the "/oic/sec/roles" resource to determine the roles that have been previously
2279 asserted. An array of credential objects shall be returned. If there are no valid certificates
2280 corresponding to the currently connected and authenticated Client's identity, then an empty
2281 array (i.e. []) shall be returned.

2282 10.4.3 OCF PKI Roots

2283 This clause intentionally left empty.

2284 10.4.4 PKI Trust Store

2285 Each Device using a certificate chained to an OCF Root CA trust anchor SHALL securely store the
2286 OCF Root CA certificates in the "oic/sec/cred" resource and SHOULD physically store this resource
2287 in a hardened memory location where the certificates cannot be tampered with.

2288 **10.4.5 Path Validation and extension processing**

2289 Devices SHALL follow the certificate path validation algorithm in clause 6 of IETF RFC 5280. In
2290 addition, the following are best practices and SHALL be adhered to by any OCF-compliant
2291 application handling digital certificates

2292 – Validity Period checking

2293 OCF-compliant applications SHALL conform to IETF RFC 5280 clauses 4.1.2.5, 4.1.2.5.1, and
2294 4.1.2.5.2 when processing the notBefore and notAfter fields in X.509 certificates. In addition,
2295 for all certificates, the notAfter value SHALL NOT exceed the notAfter value of the issuing CA.

2296 – Revocation checking

2297 Relying applications SHOULD check the revocation status for all certificates.

2298 – basicConstraints

2299 For all Root and Intermediate Certificate Authority (CA) certificates, Devices SHALL verify that
2300 the basicConstraints extension is present, flagged critical, and that the cA boolean value in the
2301 extension is TRUE. If any of these are false, the certificate chain SHALL be rejected.

2302 If the pathLenConstraint field is present, Devices will confirm the number of certificates between
2303 this certificate and the End-Entity certificate is less than or equal to pathLenConstraint. In
2304 particular, if pathLenConstraint is zero, only an End-Entity certificate can be issued by this
2305 certificate. If the pathLenConstraint field is absent, there is no limit to the chain length.

2306 For End-Entity certificates, if the basicConstraints extension is present, it SHALL be flagged
2307 critical, SHALL have a cA boolean value of FALSE, and SHALL NOT contain a
2308 pathLenConstraint ASN.1 sequence. An End-Entity certificate SHALL be rejected if a
2309 pathLenConstraint ASN.1 sequence is either present with an Integer value, or present with a
2310 null value.

2311 In order to facilitate future flexibility in OCF-compliant PKI implementations, all OCF-compliant
2312 Root CA certificates SHALL NOT contain a pathLenConstraint. This allows additional tiers of
2313 Intermediate CAs to be implemented in the future without changing the Root CA trust anchors,
2314 should such a requirement emerge.

2315 – keyUsage

2316 For all certificates, Devices shall verify that the key usage extension is present and flagged
2317 critical.

2318 For Root and Intermediate CA certificates, ONLY the keyCertSign(5) and crlSign(6) bits SHALL
2319 be asserted.

2320 For End-Entity certificates, ONLY the digitalSignature(0) and keyAgreement(4) bits SHALL be
2321 asserted.

2322 – extendedKeyUsage:

2323 Any End-Entity certificate containing the anyExtendedKeyUsage OID ("2.5.29.37.0") SHALL be
2324 rejected.

2325 OIDs for serverAuthentication ("1.3.6.1.5.5.7.3.1") and clientAuthentication ("1.3.6.1.5.5.7.3.2")
2326 are required for compatibility with various TLS implementations.

2327 At this time, an End-Entity certificate cannot be used for both Identity ("1.3.6.1.4.1.44924.1.6")
2328 and Role ("1.3.6.1.4.1.44924.1.7") purposes. Therefore, exactly one of the two OIDs SHALL be
2329 present and End-Entity certificates with ECU extensions containing both OIDs SHALL be
2330 rejected.

2331 – certificatePolicies

2332 End-Entity certificates which chain to an OCF Root CA SHOULD contain at least one
2333 PolicyIdentifierId set to the OCF Certificate Policy OID – ("1.3.6.1.4.1.51414.0.1.2")

2334 corresponding to the version of the OCF Certificate Policy under which it was issued. Additional
2335 manufacturer-specific CP OIDs may also be populated.

2336 **10.5 Device Authentication with OCF Cloud – moved to OCF Cloud Security document**

2337 This clause is intentionally left blank.

2338

2339 **11 Message Integrity and Confidentiality**

2340 **11.1 Preamble**

2341 Secured communications between Clients and Servers are protected against eavesdropping,
2342 tampering, or message replay, using security mechanisms that provide message confidentiality and
2343 integrity.

2344 **11.2 Session Protection with DTLS**

2345 **11.2.1 DTLS Protection General**

2346 Devices shall support DTLS for secured communications as defined in IETF RFC 6347. Devices
2347 using TCP shall support TLS v1.2 for secured communications as defined in IETF RFC 5246. See
2348 11.3 for a list of required and optional cipher suites for message communication.

2349 OCF Devices MUST support (D)TLS version 1.2 or greater and MUST NOT support versions 1.1
2350 or lower.

2351 Multicast session semantics are not yet defined in this version of the security document.

2352 **11.2.2 Unicast Session Semantics**

2353 For unicast messages between a Client and a Server, both Devices shall authenticate each other.
2354 See clause 10 for details on Device Authentication.

2355 Secured unicast messages between a Client and a Server shall employ a cipher suite from 11.3.
2356 The sending Device shall encrypt and authenticate messages as defined by the selected cipher
2357 suite and the receiving Device shall verify and decrypt the messages before processing them.

2358 **11.2.3 Cloud Session Semantics – moved to OCF Cloud Security document**

2359 This clause is intentionally left blank.

2360 **11.3 Cipher Suites**

2361 **11.3.1 Cipher Suites General**

2362 The cipher suites allowed for use can vary depending on the context. This clause lists the cipher
2363 suites allowed during ownership transfer and normal operation. The following RFCs provide
2364 additional information about the cipher suites used in OCF.

2365 IETF RFC 4279: Specifies use of pre-shared keys (PSK) in (D)TLS

2366 IETF RFC 4492: Specifies use of elliptic curve cryptography in (D)TLS

2367 IETF RFC 5489: Specifies use of cipher suites that use elliptic curve Diffie-Hellman (ECDHE) and
2368 PSKs

2369 IETF RFC 6655 and IETF RFC 7251: Specifies AES-CCM mode cipher suites, with ECDHE

2370 **11.3.2 Cipher Suites for Device Ownership Transfer**

2371 **11.3.2.1 Just Works Method Cipher Suites**

2372 The Just Works OTM may use the following (D)TLS cipher suites.

2373 TLS_ECDH_ANON_WITH_AES_128_CBC_SHA256,

2374 TLS_ECDH_ANON_WITH_AES_256_CBC_SHA256

2375 All Devices supporting Just Works OTM shall implement:

2376 TLS_ECDH_ANON_WITH_AES_128_CBC_SHA256 (with the value 0xFF00)

2377 All Devices supporting Just Works OTM should implement:

2378 TLS_ECDH_ANON_WITH_AES_256_CBC_SHA256 (with the value 0xFF01)

2379 **11.3.2.2 Random PIN Method Cipher Suites**

2380 The Random PIN Based OTM may use the following (D)TLS cipher suites.

2381 TLS_ECDHE_PSK_WITH_AES_128_CBC_SHA256,

2382 TLS_ECDHE_PSK_WITH_AES_256_CBC_SHA256,

2383 All Devices supporting Random Pin Based OTM shall implement:

2384 TLS_ECDHE_PSK_WITH_AES_128_CBC_SHA256

2385 **11.3.2.3 Certificate Method Cipher Suites**

2386 The Manufacturer Certificate Based OTM may use the following (D)TLS cipher suites.

2387 TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8,

2388 TLS_ECDHE_ECDSA_WITH_AES_256_CCM_8,

2389 TLS_ECDHE_ECDSA_WITH_AES_128_CCM,

2390 TLS_ECDHE_ECDSA_WITH_AES_256_CCM

2391 Using the following curve:

2392 secp256r1 (See IETF RFC 4492)

2393 All Devices supporting Manufacturer Certificate Based OTM shall implement:

2394 TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8

2395 Devices supporting Manufacturer Certificate Based OTM should implement:

2396 TLS_ECDHE_ECDSA_WITH_AES_256_CCM_8,

2397 TLS_ECDHE_ECDSA_WITH_AES_128_CCM,

2398 TLS_ECDHE_ECDSA_WITH_AES_256_CCM

2399 **11.3.3 Cipher Suites for Symmetric Keys**

2400 The following cipher suites are defined for (D)TLS communication using PSKs:

2401 TLS_ECDHE_PSK_WITH_AES_128_CBC_SHA256,

2402 TLS_ECDHE_PSK_WITH_AES_256_CBC_SHA256,

2403 TLS_PSK_WITH_AES_128_CCM_8, (* 8 OCTET Authentication tag *)

2404 TLS_PSK_WITH_AES_256_CCM_8,

2405 TLS_PSK_WITH_AES_128_CCM, (* 16 OCTET Authentication tag *)

2406 TLS_PSK_WITH_AES_256_CCM,

2407 All CCM based cipher suites also use HMAC-SHA-256 for authentication.

2408 All Devices shall implement the following:

2409 TLS_ECDHE_PSK_WITH_AES_128_CBC_SHA256,

2410

2411 Devices should implement the following:

2412 TLS_ECDHE_PSK_WITH_AES_128_CBC_SHA256,
2413 TLS_ECDHE_PSK_WITH_AES_256_CBC_SHA256,
2414 TLS_PSK_WITH_AES_128_CCM_8,
2415 TLS_PSK_WITH_AES_256_CCM_8,
2416 TLS_PSK_WITH_AES_128_CCM,
2417 TLS_PSK_WITH_AES_256_CCM

2418 **11.3.4 Cipher Suites for Asymmetric Credentials**

2419 The following cipher suites are defined for (D)TLS communication with asymmetric keys or
2420 certificates:

2421 TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8,
2422 TLS_ECDHE_ECDSA_WITH_AES_256_CCM_8,
2423 TLS_ECDHE_ECDSA_WITH_AES_128_CCM,
2424 TLS_ECDHE_ECDSA_WITH_AES_256_CCM

2425 Using the following curve:

2426 secp256r1 (See IETF RFC 4492)

2427 All Devices supporting Asymmetric Credentials shall implement:

2428 TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8

2429 All Devices supporting Asymmetric Credentials should implement:

2430 TLS_ECDHE_ECDSA_WITH_AES_256_CCM_8,
2431 TLS_ECDHE_ECDSA_WITH_AES_128_CCM,
2432 TLS_ECDHE_ECDSA_WITH_AES_256_CCM

2433 **11.3.5 Cipher suites for OCF Cloud Credentials – moved to OCF Cloud Security document**

2434 This clause is intentionally left blank.

2435

2436 **12 Access Control**

2437 **12.1 ACL Generation and Management**

2438 This clause intentionally left empty.

2439 **12.2 ACL Evaluation and Enforcement**

2440 **12.2.1 ACL Evaluation and Enforcement General**

2441 The Server enforces access control over application Resources before exposing them to the
2442 requestor. The Security Layer in the Server authenticates the requestor when access is received
2443 via the secure port. Authenticated requestors, known as the "subject" can be used to match ACL
2444 entries that specify the requestor's identity, role or may match authenticated requestors using a
2445 subject wildcard.

2446 If the request arrives over the unsecured port, the only ACL policies allowed are those that use a
2447 subject wildcard match of anonymous requestors.

2448 Access is denied if a requested Resource is not matched by an ACL entry.

2449 NOTE There are documented exceptions pertaining to Device onboarding where access to Security Virtual Resources
2450 may be granted prior to provisioning of ACL Resources.

2451 The second generation ACL (i.e. "/oic/sec/acl2") contains an array of Access Control Entries (ACE2)
2452 that employ a Resource matching algorithm that uses an array of Resource references to match
2453 Resources to which the ACE2 access policy applies. Matching consists of comparing the values of
2454 the ACE2 "resources" Property (see clause 13) to the requested Resource. Resources are matched
2455 in two ways:

- 2456 1) host reference ("href")
- 2457 2) resource wildcard ("wc").

2458 **12.2.2 Host Reference Matching**

2459 When present in an ACE2 matching element, the Host Reference (href) Property shall be used for
2460 Resource matching.

2461 – The href Property shall be used to find an exact match of the Resource name if present.

2462 **12.2.3 Resource Wildcard Matching**

2463 When present, a wildcard ("wc") expression shall be used to match multiple Resources using a
2464 wildcard Property contained in the "oic.sec.ace2.resource-ref" structure.

2465 A wildcard expression may be used to match multiple Resources using a wildcard Property
2466 contained in the "oic.sec.ace2.resource-ref" structure. The wildcard matching strings are defined
2467 in Table 14.

2468 **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.

2469 NOTE Discoverable resources appear in the "/oic/res" Resource, while non-discoverable resources may appear in other
2470 collection resources but do not appear in the /res collection.

2471 **12.2.4 Multiple Criteria Matching**

2472 If the ACE2 "resources" Property contains multiple entries, then a logical OR shall be applied for
2473 each array element. For example, if a first array element of the "resources" Property contains
2474 "href="/a/light" and the second array element of the "resources" Property contains "href="/a/led",
2475 then Resources that match either of the two "href" criteria shall be included in the set of matched
2476 Resources.

2477 Example 1 JSON for Resource matching

```
2478 {  
2479 //Matches Resources named "/x/door1" or "/x/door2"  
2480 "resources": [  
2481   {  
2482     "href": "/x/door1"  
2483   },  
2484   {  
2485     "href": "/x/door2"  
2486   },  
2487 ]  
2488 }
```

2489 Example 2 JSON for Resource matching

```
2490 {  
2491 // Matches all Resources  
2492 "resources": [  
2493   {  
2494     "wc": "*"   
2495   }  
2496 ]  
2497 }
```

2498 **12.2.5 Subject Matching using Wildcards**

2499 When the ACE subject is specified as the wildcard string "*" any requestor is matched. The OCF
2500 server may authenticate the OCF client, but is not required to.

2501 Examples: JSON for subject wildcard matching

```
2502 //matches all subjects that have authenticated and confidentiality protections in place.  
2503 "subject" : {  
2504   "conntype" : "auth-crypt"  
2505 }  
2506 //matches all subjects that have NOT authenticated and have NO confidentiality protections in place.  
2507 "subject" : {  
2508   "conntype" : "anon-clear"  
2509 }
```

2510 **12.2.6 Subject Matching using Roles**

2511 When the ACE subject is specified as a role, a requestor shall be matched if either:

- 2512 1) The requestor authenticated with a symmetric key credential, and the role is present in the
2513 "roleid" Property of the credential's entry in the "credential" Resource, or

2514 2) The requestor authenticated with a certificate, and a valid role certificate is present in the roles
2515 resource with the requestor's certificate's public key at the time of evaluation. Validating role
2516 certificates is defined in 10.3.1.

2517 **12.2.7 ACL Evaluation**

2518 **12.2.7.1 ACE2 matching algorithm**

2519 The OCF Server shall apply an ACE2 matching algorithm that matches in the following sequence:

- 2520 1) The local "/oic/sec/acl2" Resource contributes its ACE2 entries for matching.
- 2521 2) Access shall be granted when all these criteria are met:
 - 2522 a) The requestor is matched by the ACE2 "subject" Property.
 - 2523 b) The requested Resource is matched by the ACE2 "resources" Property and the requested
2524 Resource shall exist on the local Server.
 - 2525 c) The "period" Property constraint shall be satisfied.
 - 2526 d) The "permission" Property constraint shall be applied.

2527 If multiple ACE2 entries match the Resource request, the union of permissions, for all matching
2528 ACEs, defines the effective permission granted. E.g. If Perm1=CR---; Perm2=--UDN; Then UNION
2529 (Perm1, Perm2)=CRUDN.

2530 The Server shall enforce access based on the effective permissions granted.

2531 Batch requests to Resource containing Links require additional considerations when accessing the
2532 linked Resources. ACL considerations for batch request to the Atomic Measurement Resource
2533 Type are provided in clause 12.2.7.2. ACL considerations for batch request to the Collection
2534 Resource Type are provided in clause 12.2.7.3.

2535 Clause 12.2.7.4 provides ACL considerations when a new Resource is created on a Server in
2536 response to a CREATE request.

2537 **12.2.7.2 (Currently blank)**

2538 This clause intentionally left empty.

2539 **12.2.7.3 ACL considerations for a batch OCF Interface request to a Collection**

2540 This clause addresses the additional authorization processes which take place when a Server
2541 receives a batch OCF Interface request from a Client to a Collection hosted on that Server,
2542 assuming there is an ACE matching the Collection which permits the original Client request. For
2543 the purposes of this clause, the Server hosting this Collection is called the "Collection host". The
2544 additional authorization process is dependent on whether the linked Resource is hosted on the
2545 Collection host or the linked Resource is hosted on another Server:

- 2546 – For each generated request to a linked Resource hosted on the Collection host, the Collection
2547 host shall apply the ACE2 matching algorithm in clause 12.2.7.1 to determine whether the linked
2548 Resource is permitted to process the generated request, with the following clarifications:
 - 2549 – The requestor in clause 12.2.7.1 shall be the Client which sent the original Client request.
 - 2550 – The requested Resource in clause 12.2.7.1 shall be the linked Resource, which shall be
2551 matched using at least one of:
 - 2552 – a Resource Wildcard matching the linked Resource, or
 - 2553 – an exact match of the local path of the linked Resource with a "href" Property in the
2554 "resources" array in the ACE2.
 - 2555 – an exact match of the full URI of the linked Resource with a "href" Property in the
2556 "resources" array in the ACE2.

2557 NOTE The full URI of a linked Resource is obtained by concatenating the "anchor" Property of the Link, if present, and
2558 the "href" Property of the Link. The local path can then be determined from the full URI.

2559 If the linked Resource is not permitted to process the generated request, then the Collection host
2560 shall treat such cases as a linked Resource which cannot process the request when composing the
2561 aggregated response to the original Client Request, as specified for the batch OCF Interface in the
2562 ISO/IEC 30118-1:2018.

2563 **12.2.7.4 ACL Considerations on creation of a new Resource**

2564 When a new Resource is created on a Server in response to a CREATE request, there might be
2565 no ACEs permitting access to the newly created Resource. The present clause describes how the
2566 Server autonomously modifies the "/oic/sec/acl2" Resource to provide some initial authorizations
2567 for accessing the newly created Resource. The purpose of this autonomous modification is to avoid
2568 relying on the AMS update the "/oic/sec/acl2" Resource after every new Resource is created.

2569 Subsequent to a Server creating a Collection inside another Collection in response to a CREATE
2570 request from a Client, and prior to sending a response to the Client:

- 2571 – If there is an ACE with "subject" containing the UUID of the Client, and "permissions" exactly
2572 matching the CREATE, RETRIEVE, UPDATE and DELETE operations, then the Server shall
2573 autonomously add an "href" entry to "resources" with the URI of the newly created Collection.
- 2574 – Otherwise, the Server shall autonomously add an ACE with "subject" containing the UUID
2575 of the Client, "resources" containing an "href" entry with the URI of the newly created
2576 Collection, and "permissions" exactly matching the CREATE, RETRIEVE, UPDATE and
2577 DELETE operations.

2578 Subsequent to a Server creating a non-Collection Resource inside another Collection in response
2579 to a CREATE request from a Client, and prior to sending a response to the Client:

- 2580 – If there is an ACE with "subject" containing the UUID of the Client, and "permissions" exactly
2581 matching the RETRIEVE, UPDATE and DELETE operations, then the Server shall
2582 autonomously add an "href" entry to "resources" with the URI of the newly created Resource.
- 2583 – Otherwise, the Server shall autonomously add an ACE with "subject" containing the UUID
2584 of the Client, "resources" containing an "href" entry with the URI of the newly created, and
2585 "permissions" exactly matching the RETRIEVE, UPDATE and DELETE operations.

2586

2587 **13 Security Resources**

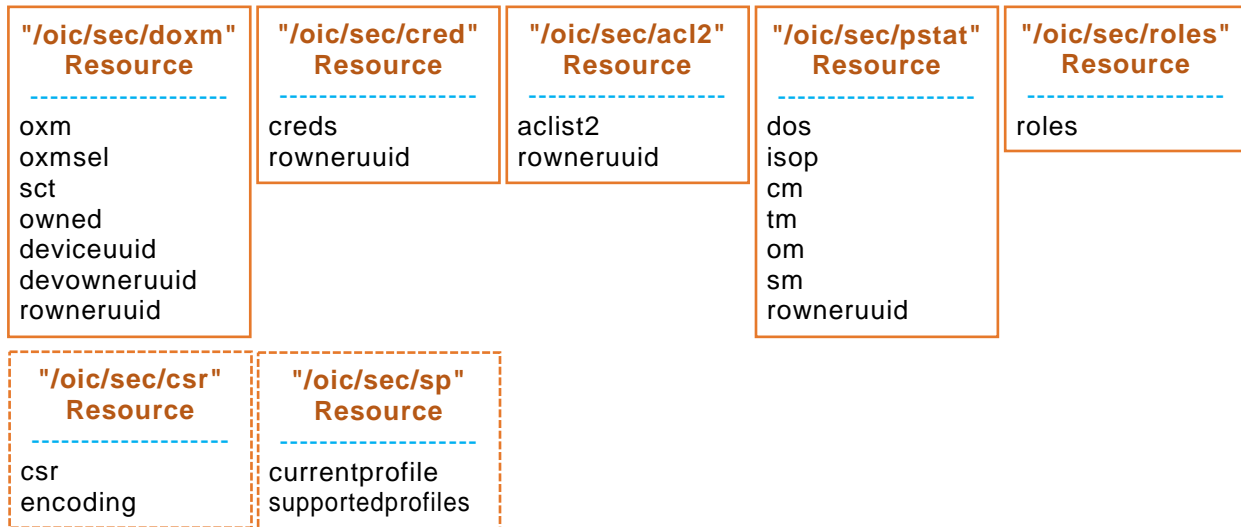
2588 **13.1 Security Resources General**

2589 OCF Security Resources are shown in Figure 21.

2590 "/oic/sec/cred" Resource and Properties are shown in Figure 22.

2591 "/oic/sec/acl2" Resource and Properties are shown in Figure 23.

2592



2593

Figure 21 – OCF Security Resources

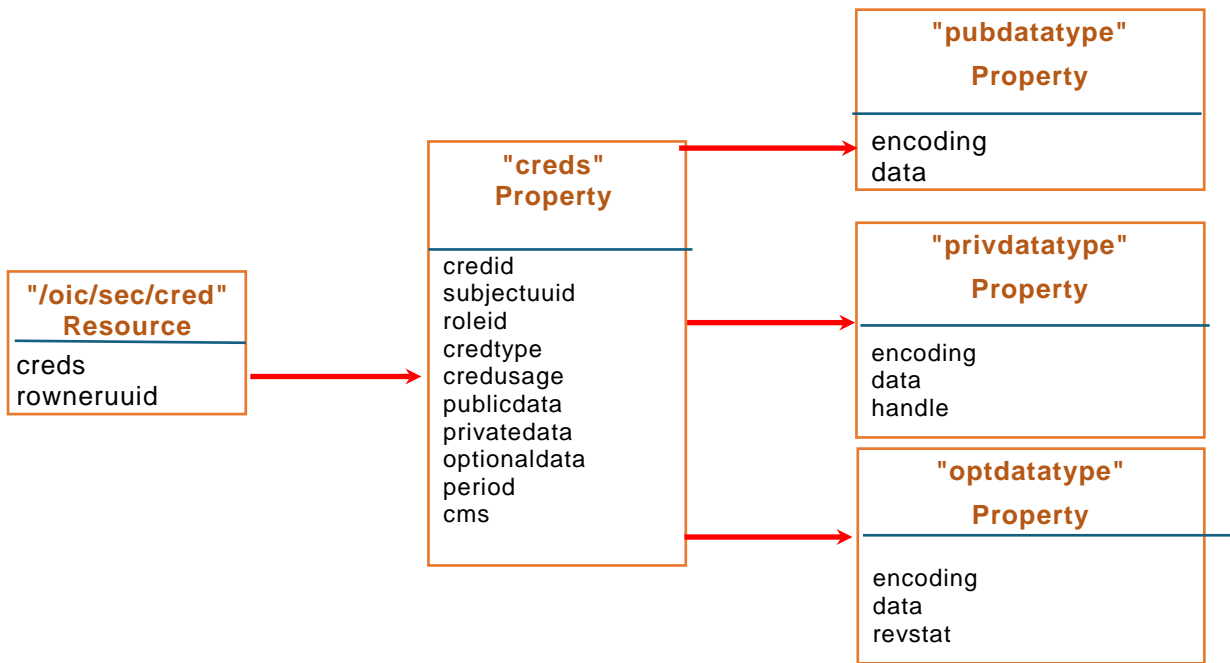


Figure 22 – "/oic/sec/cred" Resource and Properties

2594

2595

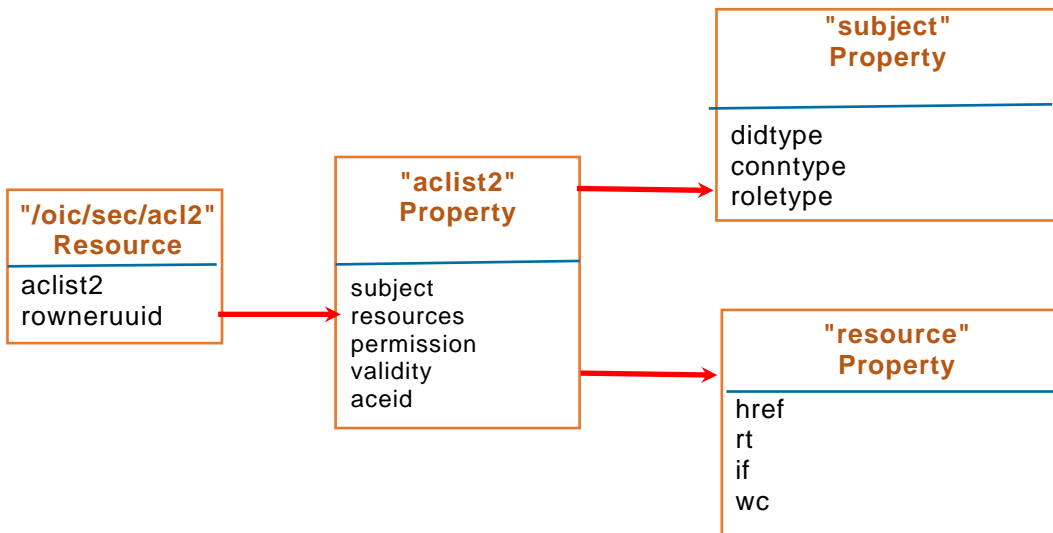


Figure 23 – "/oic/sec/acl2" Resource and Properties

2596

2597

2598 13.2 Device Owner Transfer Resource

2599 13.2.1 Device Owner Transfer Resource General

2600 The "/oic/sec/doxm" Resource contains the set of supported Device OTMs.

2601 Resource discovery processing respects the CRUDN constraints supplied as part of the security
 2602 Resource definitions contained in this document.

2603 "/oic/sec/doxm" Resource is defined in Table 15.

2604 **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

2605 Table 16 defines the Properties of the "/oic/sec/doxm" Resource.

2606 **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.
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.

2607 Table 17 defines the Properties of the "oic.sec.didtype".

2608 **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

2609 The "oxms" Property contains a list of OTM where the entries appear in the order of preference.
 2610 This Property contains the higher priority methods appearing before the lower priority methods.
 2611 The DOTS queries this list at the time of onboarding and selects the most appropriate method.

2612 OTMs consist of two parts, a URI identifying the vendor or organization and the specific method.

```

2613 <DoxmType> ::= <NSS>
2614 <NSS> ::= <Identifier> | { {<NID> "." } <NameSpaceQualifier> "." } <Method>
2615 <NID> ::= <Vendor-or-Organization>
2616 <Identifier> ::= INTEGER
2617 <NameSpaceQualifier> ::= String
2618 <Method> ::= String
2619 <Vendor-Organization> ::= String

```

2620 When an OTM successfully completes, the "owned" Property is set to "1" (TRUE). Consequently,
 2621 subsequent attempts to take ownership of the Device will fail.

2622 There are four device identifiers:

- 2623 1) "deviceuuid" Property of "/oic/sec/doxm" Resource - random DOTS-provisioned value unique
- 2624 for a given security domain, used as a device identity for access control, mapped internally to
- 2625 a device-owned credential.

2626 2) "di" Property of "/oic/d" Resource - mirroring the value of "deviceuuid" Property of
 2627 "/oic/sec/doxm" Resource.

2628 3) "piid" Property of "/oic/d" Resource - defined in ISO/IEC 30118-1:2018.

2629 4) "pi" Property of "/oic/p" Resource - defined in ISO/IEC 30118-1:2018.

2630 **13.2.2 OCF defined OTMs**

2631 Table 18 defines the Properties of the "oic.sec.doxmtype".

2632 **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 ^a .
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.oxmself" 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

^a 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.

2633 **13.3 Credential Resource**

2634 **13.3.1 Credential Resource General**

2635 The "/oic/sec/cred" Resource maintains credentials used to authenticate the Server to Clients and
 2636 support services as well as credentials used to verify Clients and support services.

2637 Multiple credential types are anticipated by the OCF framework, including pair-wise pre-shared
 2638 keys, asymmetric keys, certificates and others. The credential Resource uses a Subject UUID to
 2639 distinguish the Clients and support services it recognizes by verifying an authentication challenge.

2640 In order to provide an interface which allows management of the "creds" Array Property, the
 2641 RETRIEVE, UPDATE and DELETE operations on the "/oic/sec/cred" Resource shall behave as
 2642 follows:

- 2643 1) A RETRIEVE shall return the full Resource representation, except that any write-only Properties
 2644 shall be omitted (e.g. private key data).
- 2645 2) An UPDATE shall replace or add to the Properties included in the representation sent with the
 2646 UPDATE request, as follows:
 - 2647 a) If an UPDATE representation includes the "creds" array Property, then:
 - 2648 i) Supplied "creds" with a "credid" that matches an existing "credid" shall replace
 2649 completely the corresponding "cred" in the existing "creds" array.
 - 2650 ii) Supplied "creds" without a "credid" shall be appended to the existing "creds" array, and
 2651 a unique (to the "cred" Resource) "credid" shall be created and assigned to the new
 2652 "cred" by the Server. The "credid" of a deleted "cred" should not be reused, to improve
 2653 the determinism of the interface and reduce opportunity for race conditions.
 - 2654 iii) Supplied "creds" with a "credid" that does not match an existing "credid" shall be
 2655 appended to the existing "creds" array, using the supplied "credid".
 - 2656 iv) The rows in Table 20 corresponding to the "creds" array Property dictate the Device
 2657 States in which an UPDATE of the "creds" array Property is always rejected. If OCF
 2658 Device is in a Device State where the Access Mode in this row contains "R", then the
 2659 OCF Device shall reject all UPDATES of the "creds" array Property.
- 2660 3) A DELETE without query parameters shall remove the entire "creds" array, but shall not remove
 2661 the "/oic/sec/cred" Resource.
- 2662 4) A DELETE with one or more "credid" query parameters shall remove the "cred"(s) with the
 2663 corresponding "credid"(s) from the "creds" array.
- 2664 5) The rows in Table 20 corresponding to the "creds" array Property dictate the Device States in
 2665 which a DELETE is always rejected. If OCF Device is in a Device State where the Access Mode
 2666 in this row contains "R", then the OCF Device shall reject all DELETES.

2667 NOTE The "/oic/sec/cred" Resource's use of the DELETE operation is not in accordance with the OCF Interfaces defined
 2668 in ISO/IEC 30118-1:2018.

2669 "/oic/sec/cred" Resource is defined in Table 19.

2670 **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

2671 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.

2673 All secure Device accesses shall have a "/oic/sec/cred" Resource that protects the end-to-end
2674 interaction.

2675 The "/oic/sec/cred" Resource shall be updateable by the service named in its rowneruuid Property.

2676 ACLs naming "/oic/sec/cred" Resource should further restrict access beyond CRUDN access
2677 modes.

2678 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".

2680 Table 22 defines the Properties of "oic.sec.credusagetype".

2681 **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

2682 Table 23 defines the Properties of "oic.sec.pubdatatype".

2683 **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

2684 Table 24 defines the Properties of "oic.sec.privdatatype".

2685 **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

2686 Table 25 defines the Properties of "oic.sec.optdatatype".

2687

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

2688 Table 26 defines the Properties of "oic.sec.roletype".

2689

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.

2690 **13.3.2 Properties of the Credential Resource**2691 **13.3.2.1 Credential ID**

2692 Credential ID ("credid") is a local reference to an entry in a "creds" Property array of the
2693 "/oic/sec/cred" Resource. The SRM generates it. The "credid" Property shall be used to
2694 disambiguate array elements of the "creds" Property.

2695 **13.3.2.2 Subject UUID**

2696 The "subjectuuid" Property identifies the Device to which an entry in a "creds" Property array of the
2697 "/oic/sec/cred" Resource shall be used to establish a secure session, verify an authentication
2698 challenge-response or to authenticate an authentication challenge.

2699 A "subjectuuid" Property that matches the Server's own "deviceuuid" Property, distinguishes the
2700 array entries in the "creds" Property that pertain to this Device.

2701 The "subjectuuid" Property shall be used to identify a group to which a group key is used to protect
2702 shared data.

2703 When certificate chain is used during secure connection establishment, the "subjectuuid" Property
2704 shall also be used to verify the identity of the responder. The presented certificate chain shall be
2705 accepted, if there is a matching Credential entry on the Device that satisfies all of the following:

- 2706 – Public Data of the entry contains trust anchor (root) of the presented chain.
- 2707 – Subject UUID of the entry matches UUID in the Common Name field of the End-Entity certificate
2708 in the presented chain. If Subject UUID of the entry is set as a wildcard "*", this condition is
2709 automatically satisfied.
- 2710 – Credential Usage of the entry is "oic.sec.cred.trustca".

2711 **13.3.2.3 Role ID**

2712 The "roleid" Property identifies a role that has been granted to the credential.

2713 **13.3.2.4 Credential Type**

2714 The "credtype" Property is used to interpret several of the other Property values whose contents
2715 can differ depending on credential type. These Properties include "publicdata", "privatedata" and
2716 "optionaldata". The "credtype" Property value of "0" ("no security mode") is reserved for testing and
2717 debugging circumstances. Production deployments shall not allow provisioning of credentials of
2718 type "0". The SRM should introduce checking code that prevents its use in production deployments.

2719 **13.3.2.5 Public Data**

2720 The "publicdata" Property contains information that provides additional context surrounding the
2721 issuance of the credential. For example, it might contain information included in a certificate or
2722 response data from a CMS. It might contain wrapped data.

2723 **13.3.2.6 Private Data**

2724 The "privatedata" Property contains secret information that is used to authenticate a Device, protect
2725 data or verify an authentication challenge-response.

2726 The "privatedata" Property shall not be disclosed outside of the SRM's trusted computing perimeter.
2727 A secure element (SE) or trusted execution environment (TEE) should be used to implement the
2728 SRM's trusted computing perimeter. The privatedata contents may be referenced using a handle;
2729 for example, if used with a secure storage sub-system.

2730 **13.3.2.7 Optional Data**

2731 The "optionaldata" Property contains information that is optionally supplied, but facilitates key
2732 management, scalability or performance optimization.

2733 **13.3.2.8 Period**

2734 The "period" Property identifies the validity period for the credential. If no validity period is specified,
2735 the credential lifetime is undetermined. Constrained devices that do not implement a date-time
2736 capability shall obtain current date-time information from its CMS.

2737 **13.3.2.9 Credential Refresh Method Type Definition [Deprecated]**

2738 This clause is intentionally left blank.

2739 **13.3.2.10 Credential Usage**

2740 Credential Usage indicates to the Device the circumstances in which a credential should be used.
2741 Five values are defined:

- 2742 – "oic.sec.cred.trustca": This certificate is a trust anchor for the purposes of certificate chain
2743 validation, as defined in 10.4. OCF Server SHALL remove any "/oic/sec/cred" entries with an
2744 "oic.sec.cred.trustca" credusage upon transitioning to RFOTM. OCF Servers SHALL use
2745 "/oic/sec/cred" entries that have an "oic.sec.cred.trustca" Value of "credusage" Property only
2746 as trust anchors for post-onboarding (D)TLS session establishment in RFNOP state; these
2747 entries are not to be used for onboarding (D)TLS sessions.
- 2748 – "oic.sec.cred.cert": This "credusage" is used for certificates for which the Device possesses the
2749 private key and uses it for identity authentication in a secure session, as defined in clause 10.4.
- 2750 – "oic.sec.cred.rolecert": This "credusage" is used for certificates for which the Device possesses
2751 the private key and uses to assert one or more roles, as defined in clause 10.4.2.
- 2752 – "oic.sec.cred.mfgtrustca": This certificate is a trust anchor for the purposes of the Manufacturer
2753 Certificate Based OTM as defined in clause 7.3.6. OCF Servers SHALL use "/oic/sec/cred"

2754 entries that have an "oic.sec.cred.mfgtrustca" Value of "credusage" Property only as trust
2755 anchors for onboarding (D)TLS session establishment; these entries are not to be used for post-
2756 onboarding (D)TLS sessions.

2757 – "oic.sec.cred.mfgcert": This certificate is used for certificates for which the Device possesses
2758 the private key and uses it for authentication in the Manufacturer Certificate Based OTM as
2759 defined in clause 7.3.6.

2760 13.3.2.11 Resource Owner

2761 The Resource Owner Property allows credential provisioning to occur soon after Device onboarding
2762 before access to support services has been established. It identifies the entity authorized to
2763 manage the "/oic/sec/cred" Resource in response to Device recovery situations.

2764 13.3.3 Key Formatting

2765 13.3.3.1 Symmetric Key Formatting

2766 Symmetric keys shall have the format described in Table 27 and Table 28.

2767 **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.

2768

2769 **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.

2770 13.3.3.2 Asymmetric Keys

2771 Asymmetric key formatting is not available in this revision of the document.

2772 13.3.3.3 Asymmetric Keys with Certificate

2773 Key formatting is defined by certificate definition.

2774 13.3.3.4 Passwords

2775 Password formatting is not available in this revision of the document.

2776 13.3.4 Credential Refresh Method Details [Deprecated]

2777 This clause is intentionally left blank.

2778 13.4 Certificate Revocation List

2779 13.4.1 CRL Resource Definition [Deprecated]

2780 This clause is intentionally left blank.

2781 13.5 ACL Resources

2782 13.5.1 ACL Resources General

2783 All Resource hosted by a Server are required to match an ACL policy. ACL policies can be
2784 expressed using "/oic/sec/acl2". The subject (e.g. "deviceuid" of the Client) requesting access to

2785 a Resource shall be authenticated prior to applying the ACL check. Resources that are available
 2786 to multiple Clients can be matched using a wildcard subject. All Resources accessible via the
 2787 unsecured communication endpoint shall be matched using a wildcard subject.

2788 **13.5.2 OCF Access Control List (ACL) BNF defines ACL structures.**

2789 ACL structure in Backus-Naur Form (BNF) notation is defined in Table 29:

2790 **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>

2791 The <DeviceId> token means the requestor must possess a credential that uses <UUID> as its
 2792 identity in order to match the requestor to the <ACE> policy.

2793 The <RoleId> token means the requestor must possess a role credential with <Character> as its
 2794 role in order to match the requestor to the <ACE> policy.

2795 The <Wildcard> token "*" means any requestor is matched to the <ACE> policy, with or without
 2796 authentication.

2797 When a <SubjectId> is matched to an <ACE> policy the <ResourceRef> is used to match the <ACE>
 2798 policy to Resources.

2799 The <OIC_LINK> token contains values used to query existence of hosted Resources.

2800 The <Permission> token specifies the privilege granted by the <ACE> policy given the <SubjectId>
 2801 and <ResourceRef> matching does not produce the empty set match.

2802 Permissions are defined in terms of CREATE ("C"), RETRIEVE ("R"), UPDATE ("U"), DELETE ("D"),
 2803 NOTIFY ("N") and NIL ("-"). NIL is substituted for a permissions character that signifies the
 2804 respective permission is not granted.

2805 The empty set match result defaults to a condition where no access rights are granted.

2806 If the <Validity> token exists, the <Permission> granted is constrained to the time <Period>.
 2807 <Validity> may further be segmented into a <Recurrence> pattern where access may alternatively
 2808 be granted and rescinded according to the pattern.

2809 **13.5.3 ACL Resource**

2810 An "acl2" is a list of type "ace2".

2811 In order to provide an interface which allows management of array elements of the "aclist2"
2812 Property associated with a "/oic/sec/acl2" Resource. The RETRIEVE, UPDATE and DELETE
2813 operations on the " /oic/sec/acl2" Resource SHALL behave as follows:

- 2814 1) A RETRIEVE shall return the full Resource representation.
- 2815 2) An UPDATE shall replace or add to the Properties included in the representation sent with the
2816 UPDATE request, as follows:
- 2817 a) If an UPDATE representation includes the array Property, then:
- 2818 i) Supplied ACEs with an "aceid" that matches an existing "aceid" shall replace completely
2819 the corresponding ACE in the existing "aces2" array.
- 2820 ii) Supplied ACEs without an "aceid" shall be appended to the existing "aces2" array, and
2821 a unique (to the acl2 Resource) "aceid" shall be created and assigned to the new ACE
2822 by the Server. The "aceid" of a deleted ACE should not be reused, to improve the
2823 determinism of the interface and reduce opportunity for race conditions.
- 2824 iii) Supplied ACEs with an "aceid" that does not match an existing "aceid" shall be
2825 appended to the existing "aces2" array, using the supplied "aceid".
- 2826 iv) The rows in Table 32 corresponding to the "aclist2" array Property dictate the Device
2827 States in which an UPDATE of the "aclist2" array Property is always rejected. If OCF
2828 Device is in a Device State where the Access Mode in this row contains "R", then the
2829 OCF Device shall reject all UPDATES of the "aclist2" array Property.
- 2830 3) A DELETE without query parameters shall remove the entire "aces2" array, but shall not remove
2831 the "oic/sec/ace2" Resource.
- 2832 4) A DELETE with one or more "aceid" query parameters shall remove the ACE(s) with the
2833 corresponding "aceid"(s) from the "aces2" array.
- 2834 5) The rows in Table 32 corresponding to the "aclist2" array Property dictate the Device States in
2835 which a DELETE is always rejected. If OCF Device is in a Device State where the Access Mode
2836 in this row contains "R", then the OCF Device shall reject all DELETES.

2837 NOTE The "/oic/sec/acl2" Resource's use of the DELETE operation is not in accordance with the OCF Interfaces
2838 defined in ISO/IEC 30118-1:2018.

2839 Evaluation of local ACL Resource completes when all ACL Resource have been queried and no
2840 entry can be found for the requested Resource for the requestor – e.g. "/oic/sec/acl2" does not
2841 match the subject and the requested Resource.

2842 Table 30 defines the values of "oic.sec.crudntype".

2843

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	RETREIVE, 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

2844 "oic/sec/acl2" Resource is defined in Table 19.

2845

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

2846 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.

2848

2849 Table 33 defines the Properties of "oic.sec.ace2".

2850

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.

2851 Table 34 defines the Properties of "oic.sec.ace2.resource-ref".

2852

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.

2853 Table 35 defines the values of "oic.sec.ace2.resource-ref".

2854

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

2855 Local ACL Resources supply policy to a Resource access enforcement point within an OCF stack
 2856 instance. The OCF framework gates Client access to Server Resources. It evaluates the subject's
 2857 request using policies contained in ACL resources.

2858 Resources named in the ACL policy can be fully qualified or partially qualified. Fully qualified
 2859 Resource references include the device identifier in the href Property that identifies the remote
 2860 Resource Server that hosts the Resource. Partially qualified references mean that the local
 2861 Resource Server hosts the Resource. If a fully qualified resource reference is given, the
 2862 Intermediary enforcing access shall have a secure channel to the Resource Server and the
 2863 Resource Server shall verify the Intermediary is authorized to act on its behalf as a Resource
 2864 access enforcement point.

2865 Resource Servers should include references to Device and ACL Resources where access
2866 enforcement is to be applied. However, access enforcement logic shall not depend on these
2867 references for access control processing as access to Server Resources will have already been
2868 granted.

2869 Local ACL Resources identify a Resource Owner service that is authorized to instantiate and modify
2870 this Resource. This prevents non-terminating dependency on some other ACL Resource.
2871 Nevertheless, it should be desirable to grant access rights to ACL Resources using an ACL
2872 Resource.

2873 An ACE2 entry is considered "currently valid" if the validity period of the ACE2 entry includes the
2874 time of the request. The validity period in the ACE2 may be a recurring time period (e.g., daily from
2875 1:00-2:00). Matching the resource(s) specified in a request to the "resource" Property of the ACE2
2876 is defined in clause 12.2. For example, one way they can match is if the Resource URI in the
2877 request exactly matches one of the resource references in the ACE2 entries.

2878 A request will match an ACE2 if any of the following are true:

2879 1) The ACE2 "subject" Property is of type "oic.sec.didtype" has a UUID value that matches the
2880 "deviceuuid" Property associated with the secure session;

2881 AND the Resource of the request matches one of the "resources" Property of the ACE2
2882 "oic.sec.ace2.resource-ref";

2883 AND the ACE2 is currently valid.

2884 2) The ACE2 "subject" Property is of type "oic.sec.conntype" and has the wildcard value that
2885 matches the currently established connection type;

2886 AND the resource of the request matches one of the "resources" Property of the ACE2
2887 "oic.sec.ace2.resource-ref";

2888 AND the ACE2 is currently valid.

2889 3) When Client authentication uses a certificate credential;

2890 AND one of the "roleid" values contained in the role certificate matches the "roleid" Property of
2891 the ACE2 "oic.sec.roletype";

2892 AND the role certificate public key matches the public key of the certificate used to establish
2893 the current secure session;

2894 AND the resource of the request matches one of the array elements of the "resources" Property
2895 of the ACE2 "oic.sec.ace2.resource-ref";

2896 AND the ACE2 is currently valid.

2897 4) When Client authentication uses a certificate credential;

2898 AND the CoAP payload query string of the request specifies a role, which is member of the set
2899 of roles contained in the role certificate;

2900 AND the roleid values contained in the role certificate matches the "roleid" Property of the ACE2
2901 "oic.sec.roletype";

2902 AND the role certificate public key matches the public key of the certificate used to establish
2903 the current secure session;

2904 AND the resource of the request matches one of the "resources" Property of the ACE2
2905 "oic.sec.ace2.resource-ref";

2906 AND the ACE2 is currently valid.

2907 5) When Client authentication uses a symmetric key credential;

2908 AND one of the "roleid" values associated with the symmetric key credential used in the secure
2909 session, matches the "roleid" Property of the ACE2 "oic.sec.roletype";

2910 AND the resource of the request matches one of the array elements of the "resources" Property
2911 of the ACE2 "oic.sec.ace2.resource-ref";

2912 AND the ACE2 is currently valid.

2913 6) When Client authentication uses a symmetric key credential;

2914 AND the CoAP payload query string of the request specifies a role, which is contained in the
2915 "oic.r.cred.creds.roleid" Property of the current secure session;

2916 AND CoAP payload query string of the request specifies a role that matches the "roleid"
2917 Property of the ACE2 "oic.sec.roletype";

2918 AND the resource of the request matches one of the array elements of the "resources" Property
2919 of the ACE2 "oic.sec.ace2.resource-ref";

2920 AND the ACE2 is currently valid.

2921 A request is granted if ANY of the 'matching' ACE2 entries contain the permission to allow the
2922 request. Otherwise, the request is denied.

2923 There is no way for an ACE2 entry to explicitly deny permission to a resource. Therefore, if one
2924 Device with a given role should have slightly different permissions than another Device with the
2925 same role, they must be provisioned with different roles.

2926 The Server is required to verify that any hosted Resource has authorized access by the Client
2927 requesting access. The "/oic/sec/acl2" Resource is co-located on the Resource host so that the
2928 Resource request processing should be applied securely and efficiently. See Annex A for example.

2929 **13.6 Access Manager ACL Resource [Deprecated]**

2930 This clause is intentionally left blank.

2931 **13.7 Signed ACL Resource [Deprecated]**

2932 This clause is intentionally left blank.

2933 **13.8 Provisioning Status Resource**

2934 The "/oic/sec/pstat" Resource maintains the Device provisioning status. Device provisioning should
2935 be Client-directed or Server-directed. Client-directed provisioning relies on a Client device to
2936 determine what, how and when Server Resources should be instantiated and updated. Server-
2937 directed provisioning relies on the Server to seek provisioning when conditions dictate. Furthermore,
2938 the "/oic/sec/cred" Resource should be provisioned at ownership transfer with credentials
2939 necessary to open a secure connection with appropriate support service.

2940 "/oic/sec/pstat" Resource is defined in Table 36.

2941 **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

2942 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	Mandatory	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	TRUE (1) – "s" state is pending until all necessary changes to Device resources are complete FALSE (0) – "s" state changes are complete

2947 In all Device states:

2948 – The Device permits an authenticated and authorised Client to change the Device state of a
2949 Device by updating "pstat.dos.s" to the desired value. The allowed Device state transitions are
2950 defined in Figure 18.

2951 – Prior to updating "pstat.dos.s", the Client configures the Device to meet entry conditions for the
2952 new Device state. The SVR definitions define the entity (Client or Server) expected to perform
2953 the specific SVR configuration change to meet the entry conditions. Once the Client has
2954 configured the aspects for which the Client is responsible, it can update "pstat.dos.s". The
2955 Server then makes any changes for which the Server is responsible, including updating required
2956 SVR values, and set pstat.dos.s to the new value.

2957 – The "pstat.dos.p" Property is read-only by all Clients.

2958 – The Server sets "pstat.dos.p" to TRUE before beginning the process of updating "pstat.dos.s",
2959 and sets it back to FALSE when the "pstat.dos.s" change is completed.

2960 Any requests to update "pstat.dos.s" while "pstat.dos.p" is TRUE are denied.

2961 When Device state is RESET:

2962 – All SVR content is removed and reset to manufacturer default values.

2963 – The default manufacturer Device state is RESET.

2964 – NCRs are reset to manufacturer default values.

2965 – NCRs shall not be accessible.

2966 – After successfully processing RESET the SRM transitions to RFOTM by setting "pstat.dos.s" to
2967 RFOTM.

2968 When Device state is RFOTM:

2969 – NCRs shall not be accessible.

2970 – Before OTM is successful, the "pstat.dos.s" is read-only by unauthenticated requestors

2971 – After the OTM is successful, the "pstat.dos.s" is read-write by authorized requestors.

2972 – The negotiated Device OC is used to create an authenticated session over which the DOTS
2973 directs the Device state to transition to RFPRO.

- 2974 – If an authenticated session cannot be established the ownership transfer session should be
2975 disconnected and SRM sets back the Device state to RESET state.
- 2976 – Ownership transfer session, especially Random PIN OTM, should not exceed 60 seconds, the
2977 SRM asserts the OTM failed, should be disconnected, and transitions to RESET
2978 ("/pstat.dos.s"=RESET).
- 2979 – The DOTS UPDATES the "devowneruuid" Property in the "/oic/sec/doxm" Resource to a non-
2980 nil UUID value. The DOTS (or other authorized client) can update it multiple times while in
2981 RFOTM. It is not updatable while in other device states except when the Device state returns
2982 to RFOTM through RESET.
- 2983 – The DOTS can have additional provisioning tasks to perform while in RFOTM. When done, the
2984 DOTS UPDATES the "owned" Property in the "/oic/sec/doxm" Resource to "true".
- 2985 When Device state is RFPRO:
- 2986 – The "pstat.dos.s" is read-only by unauthorized requestors and read-write by authorized
2987 requestors.
- 2988 – NCRs shall not be accessible, except for Easy Setup Resources, if supported.
- 2989 – The OCF Server may re-create NCRs.
- 2990 – An authorized Client may provision SVRs as needed for normal functioning in RFNOP.
- 2991 – An authorized Client may perform consistency checks on SVRs to determine which shall be re-
2992 provisioned.
- 2993 – Failure to successfully provision SVRs may trigger a state change to RESET. For example, if
2994 the Device has already transitioned from SRESET but consistency checks continue to fail.
- 2995 – The authorized Client sets the "/pstat.dos.s"=RFNOP.
- 2996 When Device state is RFNOP:
- 2997 – The "/pstat.dos.s" Property is read-only by unauthorized requestors and read-write by
2998 authorized requestors.
- 2999 – NCRs, SVRs and core Resources are accessible following normal access processing.
- 3000 – An authorized may transition to RFPRO. Only the Device owner may transition to SRESET or
3001 RESET.
- 3002 When Device state is SRESET:
- 3003 – NCRs shall not be accessible. The integrity of NCRs may be suspect but the SRM doesn't
3004 attempt to access or reference them.
- 3005 – SVR integrity is not guaranteed, but access to some SVR Properties is necessary. These
3006 include "devowneruuid" Property of the "/oic/sec/doxm" Resource,
3007 "creds":[{..., {"subjectuuid":<devowneruuid>}, ...}] Property of the "/oic/sec/cred" Resource and
3008 "pstat.dos.s" "/oic/sec/pstat" Resource.
- 3009 – The certificates that identify and authorize the Device owner are sufficient to re-create
3010 minimalist "/oic/sec/cred" and "/oic/sec/doxm" Resources enabling Device owner control of
3011 SRESET. If the SRM can't establish these Resources, then it will transition to RESET state.
- 3012 – An authorized Client performs SVR consistency checks. The authorized Client can provision
3013 SVRs as needed to ensure they are available for continued provisioning in RFPRO or for normal
3014 functioning in RFNOP.
- 3015 – The authorized Device owner can avoid entering RESET state and RFOTM by UPDATING
3016 "pstat.dos.s" with RFPRO or RFNOP values.
- 3017 – ACLs on SVR are presumed to be invalid. Access authorization is granted according to Device
3018 owner privileges only.

3019 – The SRM asserts a Client-directed operational mode (e.g. "/pstat.om"=CLIENT_DIRECTED).
 3020 The provisioning mode type is a 16-bit mask enumerating the various Device provisioning modes.
 3021 "{ProvisioningMode}" should be used in this document to refer to an instance of a provisioning
 3022 mode without selecting any particular value.

3023 "oic.sec.dpmttype" is defined in Table 39.

3024 **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

3025 Table 40 and Table 41 define the values of "oic.sec.dpmttype".

3026 **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)

3027 **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

3028 The provisioning operation mode type is an 8-bit mask enumerating the various provisioning
 3029 operation modes.

3030 "oic.sec.pomtype" is defined in Table 42.

3031 **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

3032 Table 43 defines the values of "oic.sec.pomtype".

3033

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

3034 **13.9 Certificate Signing Request Resource**

3035 The "/oic/sec/csr" Resource is used by a Device to provide its desired identity, public key to be
 3036 certified, and a proof of possession of the corresponding private key in the form of a IETF RFC
 3037 2986 PKCS#10 Certification Request. If the Device supports certificates (i.e. the "sct" Property of
 3038 "/oic/sec/doxm" Resource has a 1 in the 0x8 bit position), the Device shall have a "/oic/sec/csr"
 3039 Resource.

3040 "/oic/sec/csr" Resource is defined in Table 44.

3041

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

3042 Table 45 defines the Properties of "/oic/sec/csr".

3043

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

3044 The Device chooses which public key to use, and may optionally generate a new key pair for this
 3045 purpose.

3046 In the CSR, the Common Name component of the Subject Name shall contain a string of the format
 3047 "uuid:X" where X is the Device's requested UUID in the format defined by IETF RFC 4122. The
 3048 Common Name, and other components of the Subject Name, may contain other data. If the Device
 3049 chooses to include additional information in the Common Name component, it shall delimit it from
 3050 the UUID field by white space, a comma, or a semicolon.

3051 If the Device does not have a pre-provisioned key pair to use, but is capable and willing to generate
3052 a new key pair, the Device may begin generation of a key pair as a result of a RETRIEVE of this
3053 resource. If the Device cannot immediately respond to the RETRIEVE request due to time required
3054 to generate a key pair, the Device shall return an "operation pending" error. This indicates to the
3055 Client that the Device is not yet ready to respond, but will be able at a later time. The Client should
3056 retry the request after a short delay.

3057 **13.10 Roles Resource**

3058 The "roles" Resource maintains roles that have been asserted with role certificates, as described
3059 in clause 10.4.2. Asserted roles have an associated public key, i.e., the public key in the role
3060 certificate. Servers shall only grant access to the roles information associated with the public key
3061 of the Client. The roles Resource should be viewed as an extension of the (D)TLS session state.
3062 See 10.4.2 for how role certificates are validated.

3063 The roles Resource shall be created by the Server upon establishment of a secure (D)TLS session
3064 with a Client, if is not already created. The roles Resource shall only expose a secured OCF
3065 Endpoint in the "/oic/res" response. A Server shall retain the roles Resource at least as long as the
3066 (D)TLS session exists. A Server shall retain each certificate in the roles Resource at least until the
3067 certificate expires or the (D)TLS session ends, whichever is sooner. The requirements of clause
3068 10.3 and 10.4.2 to validate a certificate's time validity at the point of use always apply. A Server
3069 should regularly inspect the contents of the roles resource and purge contents based on a policy it
3070 determines based on its resource constraints. For example, expired certificates, and certificates
3071 from Clients that have not been heard from for some arbitrary period of time could be candidates
3072 for purging.

3073 The OCF namespace ("oic.role.*") is restricted to OCF-defined roles. "oic.role.owner" is an OCF-
3074 defined Role that is intended to provide Resource Owner privileges to multiple Clients in a scalable
3075 way. Servers shall grant access to perform all supported operations in the current Device state
3076 (see clause 8) on all supported SVRs regardless of ACL configuration the Clients asserting
3077 "oic.role.owner" Role. Servers shall reject assertion of any Role, which starts with "oic.role.", but
3078 is not one of the following Roles:

3079 – "oic.role.owner"

3080 The "roles" Resource is implicitly created by the Server upon establishment of a (D)TLS session.
3081 In more detail, the RETRIEVE, UPDATE and DELETE operations on the roles Resource shall
3082 behave as follows. Unlisted operations are implementation specific and not reliable.

- 3083 1) A RETRIEVE request shall return all previously asserted roles associated with the currently
3084 connected and authenticated Client's identity. RETRIEVE requests with a "credid" query
3085 parameter is not supported; all previously asserted roles associated with the currently
3086 connected and authenticated Client's identity are returned.
- 3087 2) An UPDATE request that includes the "roles" Property shall replace or add to the Properties
3088 included in the array as follows:
 - 3089 a) If either the "publicdata" or the "optionaldata" are different than the existing entries in the
3090 "roles" array, the entry shall be added to the "roles" array with a new, unique "credid" value.
 - 3091 b) If both the "publicdata" and the "optionaldata" match an existing entry in the "roles" array,
3092 the entry shall be considered to be the same. The Server shall reply with a 2.04 Changed
3093 response and a duplicate entry shall not be added to the array.
 - 3094 c) The "credid" Property is optional in an UPDATE request and if included, it may be ignored
3095 by the Server. The Server shall assign a unique "credid" value for every entry of the "roles"
3096 array.

3097 3) A DELETE request without a "credid" query parameter shall remove all entries from the
 3098 "/oic/sec/roles" resource array corresponding to the currently connected and authenticated
 3099 Client's identity.

3100 4) A DELETE request with a "credid" query parameter shall remove only the entries of the
 3101 "/oic/sec/roles" resource array corresponding to the currently connected and authenticated
 3102 Client's identity and where the corresponding "credid" matches the entry.

3103 NOTE The "/oic/sec/roles" Resource's use of the DELETE operation is not in accordance with the OCF Interfaces
 3104 defined in ISO/IEC 30118-1:2018.

3105 See clause 8 for restrictions on the states in which this Resource may be modified.

3106 "/oic/sec/roles" Resource is defined in Table 46.

3107 **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

3108 Table 47 defines the Properties of "/oic/sec/roles".

3109 **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

3110 Because "/oic/sec/roles" shares the "oic.sec.cred" schema with "/oic/sec/cred", "subjectuid" is a required Property.
 3111 However, "subjectuid" is not used in a role certificate. Therefore, a Device may ignore the "subjectuid" Property if the
 3112 Property is contained in an UPDATE request to the "/oic/sec/roles" Resource.

3113 **13.11 Account Resource – moved to OCF Cloud Security document**

3114 This clause is intentionally left blank.

3115 **13.12 Account Session Resource – moved to OCF Cloud Security document**

3116 This clause is intentionally left blank.

3117 **13.13 Account Token Refresh Resource – moved to OCF Cloud Security document**

3118 This clause is intentionally left blank.

3119 **13.14 Security Virtual Resources (SVRs) and Access Policy**

3120 The SVRs expose the security-related Properties of the Device.

3121 Granting access requests (RETRIEVE, UPDATE, DELETE, etc.) for these SVRs to unauthenticated
 3122 (anonymous) Clients could create privacy or security concerns.

3123 For example, when the Device onboarding State is RFOTM, it is necessary to grant requests for
 3124 the "/oic/sec/doxm" Resource to anonymous requesters, so that the Device can be discovered and
 3125 onboarded by an OBT. Subsequently, it might be preferable to deny requests for the
 3126 "/oic/sec/doxm" Resource to anonymous requesters, to preserve privacy.

3127 **13.15 SVRs, Discoverability and OCF Endpoints**

3128 All implemented SVRs shall be "discoverable" (reference ISO/IEC 30118-1:2018, Policy Parameter
 3129 clause 7.8.2.1.2).

3130 All implemented discoverable SVRs shall expose a Secure OCF Endpoint (e.g. CoAPS) (reference
3131 ISO/IEC 30118-1:2018, clause 10).

3132 The "/oic/sec/doxm" Resource shall expose an Unsecure OCF Endpoint (e.g. CoAP) in RFOTM
3133 (reference ISO/IEC 30118-1:2018, clause 10).

3134 **13.16 Additional Privacy Consideration for Core Resources**

3135 Unique immutable identifiers are a privacy consideration due to their potential for being used as a
3136 tracking mechanism. These include the following Resources and Properties:

3137 – "/oic/d" Resource containing the "piid" Property.

3138 – "/oic/p" Resource containing the "pi" Property.

3139 These identifiers are unique values that are visible at various times throughout the Device lifecycle
3140 by anonymous requestors. This implies any Client Device, including those with malicious intent,
3141 are able to reliably obtain identifiers useful for building a log of activity correlated with a specific
3142 Platform and Device.

3143 The "di" Property in the "/oic/d" Resource shall mirror that of the "deviceuuid" Property of the
3144 "/oic/sec/doxm" Resource. The DOTS should provision an ACL policy that restricts access to the
3145 "/oic/d" resource such that only authenticated Clients are able to obtain the "di" Property of "/oic/d"
3146 Resource. See clause 13.1 for deviceuuid Property lifecycle requirements.

3147 Servers should expose a temporary, non-repeated, "piid" Property of "/oic/d" Resource Value upon
3148 entering RESET Device state. Servers shall expose a persistent value via the "piid" Property of
3149 "/oic/d" Property when the DOTS sets "devowneruuid" Property to a non-nil-UUID value. The DOTS
3150 should provision an ACL policy on the "/oic/d" Resource such that only authenticated Clients are
3151 able to obtain the "piid" Property of "/oic/d" Resource

3152 Servers should expose a temporary, non-repeated, "pi" Property value upon entering RESET
3153 Device state. Servers shall expose a persistent value via the "pi" Property of the "/oic/p" Resource
3154 when the DOTS sets "devowneruuid" Property to a non-nil-UUID value. The DOTS should provision
3155 an ACL policy on the "/oic/p" Resource such that only authenticated Clients are able to obtain the
3156 "pi" Property.

3157 Table 48 depicts Core Resource Properties Access Modes given various Device States.

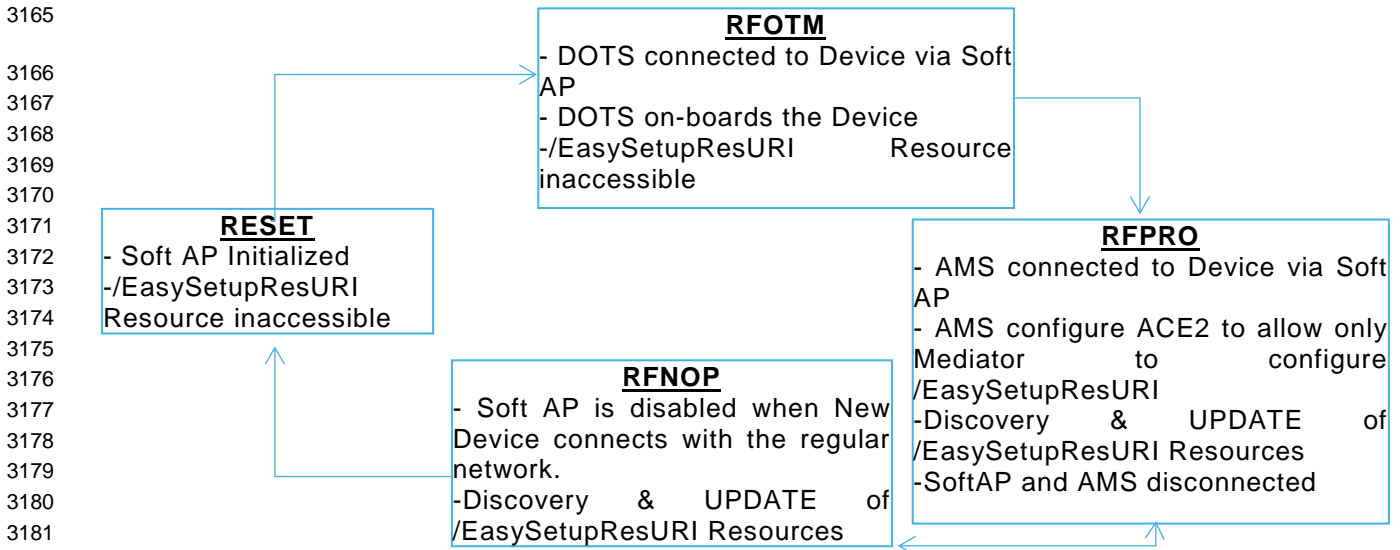
3158 **Table 48 – Core Resource Properties Access Modes given various Device States**

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.

3159 **13.17 Easy Setup Resource Device State**

3160 This clause only applies to a new Device that uses Easy Setup for ownership transfer as defined
 3161 in OCF Wi-Fi Easy Setup. Easy Setup has no impact to new Devices that have a different way of
 3162 connecting to the network i.e. DOTS and AMS don't use a Soft AP to connect to non-Easy Setup
 3163 Devices.

3164 Figure 24 shows an example of Soft AP and Easy Setup Resource in different Device states.



3182 **Figure 24 – Example of Soft AP and Easy Setup Resource in different Device states**

3183 Device enters RFOTM Device state, Soft AP may be accessible in RFOTM and RFPRO Device's
 3184 state.

3185 While it is reasonable for a user to expect that power cycling a new Device will turn on the Soft AP
 3186 for Easy Setup during the initial setup, since that is potentially how it behaved on first boot, it is a
 3187 security risk to make this the default behaviour of a device that remains unenrolled beyond a
 3188 reasonable period after first boot.

3189 Therefore, the Soft AP for Easy Setup has several requirements to improve security:

- 3190 – Time availability of Easy Setup Soft AP should be minimised, and shall not exceed 30 minutes
 3191 after Device factory reset RESET or first power boot, or when user initiates the Soft AP for Easy
 3192 Setup.
- 3193 – If a new Device tried and failed to complete Easy Setup Enrolment immediately following the
 3194 first boot, or after a factory reset, it may turn the Easy Setup Soft AP back on automatically for
 3195 another 30 minutes upon being power cycled, provided that the power cycle occurs within 3
 3196 hours of first boot or the most recent factory reset. If the user has initiated the Easy Setup Soft
 3197 AP directly without a factory reset, it is not necessary to turn it back on if it was on immediately
 3198 prior to power cycle, because the user obviously knows how to initiate the process manually.
- 3199 – After 3 hours from first boot or factory reset without successfully enrolling the device, the Soft
 3200 AP should not turn back on for Easy Setup until another factory reset occurs, or the user initiates
 3201 the Easy Setup Soft AP directly.
- 3202 – Easy Setup Soft AP may stay enabled during RFNOP, until the Mediator instructs the new
 3203 Device to connect to the Enroller.
- 3204 – The Easy Setup Soft AP shall be disabled when the new Device successfully connects to the
 3205 Enroller.

3206 – Once a new Device has successfully connected to the Enroller, it shall not turn the Easy Setup
3207 Soft AP back on for Easy Setup Enrolment again unless the Device is factory reset, or the user
3208 initiates the Easy Setup Soft AP directly.

3209 – Just Works OTM shall not be enabled on Devices which support Easy Setup.

3210 – The Soft AP shall be secured (e.g. shall not expose an open AP).

3211 – The Soft AP shall support a passphrase for connection by the Mediator, and the passphrase
3212 shall be between and 8 and 64 ASCII printable characters. The passphrase may be printed on
3213 a label, sticker, packaging etc., and may be entered by the user into the Mediator device.

3214 – The Soft AP should not use a common passphrase across multiple Devices. Instead, the
3215 passphrase may be sufficiently unique per device, to prevent guessing of the passphrase by an
3216 attacker with knowledge of the Device type, model, manufacturer, or any other information
3217 discoverable through Device's exposed interfaces.

3218 The Enrollee shall support WPA2 security (i.e. shall list WPA2 in the "swat" Property of the
3219 "/example/WiFiConfResURI" Resource), for potential selection by the Mediator in connecting the
3220 Enrollee to the Enroller. The Mediator should select the best security available on the Enroller, for
3221 use in connecting the Enrollee to the Enroller.

3222 The Enrollee may not expose any interfaces (e.g. web server, debug port, NCRs, etc.) over the
3223 Soft AP, other than SVRs, and Resources required for Wi-Fi Easy Setup.

3224 The "/example/EasySetupResURI" Resource should not be discoverable in RFOTM or SRESET
3225 state. After ownership transfer process is completed with the DOTS, and the Device enters in
3226 RFPRO Device state, the "/example/EasySetupResURI" may be Discoverable.

3227 The OTM CoAPS session may be used by Mediator for connection over Soft AP for ownership
3228 transfer and initial Easy Setup provisioning. SoftAP or regular network connection may be used by
3229 AMS for "/oic/sec/acl2" Resource provisioning in RFPRO state. The CoAPS session authentication
3230 and encryption is already defined in the Security spec.

3231 In RFPRO state, AMS is expected to configure ACL2 Resource on the Device with ACE2 for
3232 following Resources to be only configurable by the Mediator with permission to UPDATE or
3233 RETRIEVE access:

3234 – "/example/EasySetupResURI"
3235 – "/example/WifiConfResURI"
3236 – "/example/DevConfResURI"

3237 An ACE2 granting RETRIEVE or UPDATE access to the Easy Setup Resource

```

3238 {
3239     "subject": { "uuid": "<insert-UUID-of-Mediator>" },
3240     "resources": [
3241         { "href": "/example/EasySetupResURI" },
3242         { "href": "/example/WiFiConfResURI" },
3243         { "href": "/example/DevConfResURI" },
3244     ],
3245     "permission": 6 // RETRIEVE (2) or UPDATE and RETRIEVE(6)
3246 }

```

3247 ACE2 may be re-configured after Easy Setup process. These ACE2s should be installed prior to
3248 the Mediator performing any RETRIEVE/UPDATE operations on these Resources.

3249 In RFPRO or RFNOP, the Mediator should discover /EasySetupResURI Resources and UPDATE
3250 these Resources. The Mediator may UPDATE /EasySetupResURI resources in RFNOP Device
3251 state.

3252 A Mediator shall be hosted on an OCF Device.

3253 **14 Security Hardening Guidelines/ Execution Environment Security**

3254 **14.1 Preamble**

3255 This is an informative clause. Many TGs in OCF have security considerations for their protocols
3256 and environments. These security considerations are addressed through security mechanisms
3257 specified in the security documents for OCF. However, effectiveness of these mechanisms depends
3258 on security robustness of the underlying hardware and software Platform. This clause defines the
3259 components required for execution environment security.

3260 **14.2 Execution Environment Elements**

3261 **14.2.1 Execution Environment Elements General**

3262 Execution environment within a computing Device has many components. To perform security
3263 functions in a robustness manner, each of these components has to be secured as a separate
3264 dimension. For instance, an execution environment performing AES cannot be considered secure
3265 if the input path entering keys into the execution engine is not secured, even though the partitions
3266 of the CPU, performing the AES encryption, operate in isolation from other processes. Different
3267 dimensions referred to as elements of the execution environment are listed below. To qualify as a
3268 secure execution environment (SEE), the corresponding SEE element must qualify as secure.

- 3269 – (Secure) Storage
- 3270 – (Secure) Execution engine
- 3271 – (Trusted) Input/output paths
- 3272 – (Secure) Time Source/clock
- 3273 – (Random) number generator
- 3274 – (Approved) cryptographic algorithms
- 3275 – Hardware Tamper (protection)

3276 NOTE Software security practices (such as those covered by OWASP) are outside scope of this document, as
3277 development of secure code is a practice to be followed by the open source development community. This document will
3278 however address the underlying Platform assistance required for executing software. Examples are secure boot and
3279 secure software upgrade.

3280 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,
3281 14.2.7.

3282 **14.2.2 Secure Storage**

3283 **14.2.2.1 Secure Storage General**

3284 Secure storage refers to the physical method of housing sensitive or confidential data ("Sensitive
3285 Data"). Such data could include but not be limited to symmetric or asymmetric private keys,
3286 certificate data, OCF Security Domain access credentials, or personal user information. Sensitive
3287 Data requires that its integrity be maintained, whereas Critical Sensitive Data requires that both its
3288 integrity and confidentiality be maintained.

3289 It is strongly recommended that IoT Device makers provide reasonable protection for Sensitive
3290 Data so that it cannot be accessed by unauthorized Devices, groups or individuals for either
3291 malicious or benign purposes. In addition, since Sensitive Data is often used for authentication and
3292 encryption, it must maintain its integrity against intentional or accidental alteration.

3293 A partial list of Sensitive Data is outlined in Table 49:

Table 49 – Examples of Sensitive Data

Data	Integrity protection	Confidentiality protection
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

3295 Exact method of protection for secure storage is implementation specific, but typically combinations
3296 of hardware and software methods are used.

3297 **14.2.2.2 Hardware Secure Storage**

3298 Hardware secure storage is recommended for use with critical Sensitive Data such as symmetric
3299 and asymmetric private keys, access credentials, and personal private data. Hardware secure
3300 storage most often involves semiconductor-based non-volatile memory ("NVRAM") and includes
3301 countermeasures for protecting against unauthorized access to Critical Sensitive Data.

3302 Hardware-based secure storage not only stores Sensitive Data in NVRAM, but also provides
3303 protection mechanisms to prevent the retrieval of Sensitive Data through physical and/or electronic
3304 attacks. It is not necessary to prevent the attacks themselves, but an attempted attack should not
3305 result in an unauthorized entity successfully retrieving Sensitive Data.

3306 Protection mechanisms should provide JIL Moderate protection against access to Sensitive Data
3307 from attacks that include but are not limited to:

- 3308 1) Physical decapping of chip packages to optically read NVRAM contents
- 3309 2) Physical probing of decapped chip packages to electronically read NVRAM contents
- 3310 3) Probing of power lines or RF emissions to monitor voltage fluctuations to discern the bit patterns
3311 of Critical Sensitive Data
- 3312 4) Use of malicious software or firmware to read memory contents at rest or in transit within a
3313 microcontroller
- 3314 5) Injection of faults that induce improper Device operation or loss or alteration of Sensitive Data

3315 **14.2.2.3 Software Storage**

3316 It is generally NOT recommended to rely solely on software and unsecured memory to store
3317 Sensitive Data even if it is encrypted. Critical Sensitive Data such as authentication and encryption
3318 keys should be housed in hardware secure storage whenever possible.

3319 Sensitive Data stored in volatile and non-volatile memory shall be encrypted using acceptable
3320 algorithms to prevent access by unauthorized parties through methods described in 14.2.2.2.

3321 **14.2.2.4 Additional Security Guidelines and Best Practices**

3322 Some general practices that can help ensure that Sensitive Data is not compromised by various
3323 forms of security attacks:

- 3324 1) FIPS Random Number Generator ("RNG") – Insufficient randomness or entropy in the RNG
3325 used for authentication challenges can substantially degrade security strength. For this reason,
3326 it is recommended that a FIPS 800-90A-compliant RNG with a certified noise source be used
3327 for all authentication challenges.
- 3328 2) Secure download and boot – To prevent the loading and execution of malicious software, where
3329 it is practical, it is recommended that Secure Download and Secure Boot methods that
3330 authenticate a binary's source as well as its contents be used.
- 3331 3) Deprecated algorithms – Algorithms included but not limited to the list below are considered
3332 insecure and shall not be used for any security-related function:
 - 3333 a) SHA-1
 - 3334 b) MD5
 - 3335 c) RC4
 - 3336 d) RSA 1024
- 3337 4) Encrypted transmission between blocks or components – Even if critical Sensitive Data is
3338 stored in Secure Storage, any use of that data that requires its transmission out of that Secure
3339 Storage should be encrypted to prevent eavesdropping by malicious software within an
3340 MCU/MPU.
- 3341 5) It is recommended to avoid using wildcard in Subject Id ("*"), when setting up "/oic/sec/cred"
3342 Resource entries, since this opens up an identity spoofing opportunity.
- 3343 6) Device vendor understands that it is the Device vendor's responsibility to ensure the Device
3344 meets security requirements for its intended uses. As an example, IoTivity is a reference
3345 implementation intended to be used as a basis for a product, but IoTivity has not undergone
3346 3rd party security review, penetration testing, etc. Any Device based on IoTivity should undergo
3347 appropriate penetration testing and security review prior to sale or deployment.
- 3348 7) Device vendor agrees to publish the expected support lifetime for the Device to OCF and to
3349 consumers. Changes should be made to a public and accessible website. Expectations should
3350 be clear as to what will be supported and for how long the Device vendor expects to support
3351 security updates to the software, operating system, drivers, networking, firmware and hardware
3352 of the device.
- 3353 8) Device vendor has not implemented test or debug interfaces on the Device which are operable
3354 or which can be enabled which might present an attack vector on the Device which circumvents
3355 the interface-level security or access policies of the Device.
- 3356 9) Device vendor understands that if an application running on the Device has access to
3357 cryptographic elements such as the private keys or Ownership Credential, then those elements
3358 have become vulnerable. If the Device vendor is implementing a Bridge, an OBT, or a Device
3359 with access to the Internet beyond the local network, the execution of critical functions should
3360 take place within a Trusted or Secure Execution Environment (TEE/SEE).
- 3361 10) Any PINs or fixed passphrases used for onboarding, Wi-Fi Easy Setup, SoftAP management or
3362 access, or other security-critical function, should be sufficiently unique (do not duplicate
3363 passphrases. The creation of these passphrases or PINS should not be algorithmically
3364 deterministic nor should they use insufficient entropy in their creation.
- 3365 11) Ensure that there are no remaining "VENDOR_TODO" items in the source code.

3366 12) If the implementation of this document uses the "Just Works" onboarding method, understand
3367 that there is a man-in-the-middle vulnerability during the onboarding process where a malicious
3368 party could intercept messages between the device being onboarded and the OBT and could
3369 persist, acting as an intermediary with access to message traffic, during the lifetime of that
3370 onboarded device. The recommended best practice would be to use an alternate ownership
3371 transfer method (OTM) instead of "Just Works".

3372 13) It is recommended that at least one static and dynamic analysis tool¹ be applied to any
3373 proposed major production release of the software before its release, and any vulnerabilities
3374 resolved.

3375 **14.2.3 Secure execution engine**

3376 Execution engine is the part of computing Platform that processes security functions, such as
3377 cryptographic algorithms or security protocols (e.g. DTLS). Securing the execution engine requires
3378 the following

3379 – Isolation of execution of sensitive processes from unauthorized parties/ processes. This
3380 includes isolation of CPU caches, and all of execution elements that needed to be considered
3381 as part of trusted (crypto) boundary.

3382 – Isolation of data paths into and out of execution engine. For instance, both unencrypted but
3383 sensitive data prior to encryption or after decryption, or cryptographic keys used for
3384 cryptographic algorithms, such as decryption or signing. See clause 14.2.4 for more details.

3385 **14.2.4 Trusted input/output paths**

3386 Paths/ ports used for data entry into or export out of trusted/ crypto-boundary needs to be protected.
3387 This includes paths into and out secure execution engine and secure memory.

3388 Path protection can be both hardware based (e.g. use of a privileged bus) or software based (using
3389 encryption over an untrusted bus).

3390 **14.2.5 Secure clock**

3391 Many security functions depend on time-sensitive credentials. Examples are time stamped
3392 Kerberos tickets, OAuth tokens, X.509 certificates, OSCP response, software upgrades, etc. Lack
3393 of secure source of clock can mean an attacker can modify the system clock and fool the validation
3394 mechanism. Thus an SEE needs to provide a secure source of time that is protected from tampering.
3395 Trustworthiness from security robustness standpoint is not the same as accuracy. Protocols such
3396 as NTP can provide rather accurate time sources from the network, but are not immune to attacks.
3397 A secure time source on the other hand can be off by seconds or minutes depending on the time-
3398 sensitivity of the corresponding security mechanism. Secure time source can be external as long
3399 as it is signed by a trusted source and the signature validation in the local Device is a trusted
3400 process (e.g. backed by secure boot).

3401 **14.2.6 Approved algorithms**

3402 An important aspect of security of the entire ecosystem is the robustness of publicly vetted and
3403 peer-reviewed (e.g. NIST-approved) cryptographic algorithms. Security is not achieved by
3404 obscurity of the cryptographic algorithm. To ensure both interoperability and security, not only
3405 widely accepted cryptographic algorithms must be used, but also a list of approved cryptographic
3406 functions must be specified explicitly. As new algorithms are NIST approved or old algorithms are
3407 deprecated, the list of approved algorithms must be maintained by OCF. All other algorithms (even
3408 if they deemed stronger by some parties) must be considered non-approved.

3409 The set of algorithms to be considered for approval are algorithms for

3410 – Hash functions

¹ A general discussion of analysis tools can be found here: <https://www.ibm.com/developerworks/library/se-static/>

- 3411 – Signature algorithms
 - 3412 – Encryption algorithms
 - 3413 – Key exchange algorithms
 - 3414 – Pseudo Random functions (PRF) used for key derivation
- 3415 This list will be included in this or a separate security robustness rules document and must be
3416 followed for all security specifications within OCF.

3417 **14.2.7 Hardware tamper protection**

3418 Various levels of hardware tamper protection exist. We borrow FIPS 140-2 terminology (not
3419 requirements) regarding tamper protection for cryptographic module

- 3420 – Production-grade (lowest level): this means components that include conformal sealing coating
3421 applied over the module’s circuitry to protect against environmental or other physical damage.
3422 This does not however require zeroization of secret material during physical maintenance. This
3423 definition is borrowed from FIPS 140-2 security level 1.
- 3424 – Tamper evident/proof (mid-level), This means the Device shows evidence (through covers,
3425 enclosures, or seals) of an attempted physical tampering. This definition is borrowed from FIPS
3426 140-2 security level 2.
- 3427 – Tamper resistance (highest level), this means there is a response to physical tempering that
3428 typically includes zeroization of sensitive material on the module. This definition is borrowed
3429 from FIPS 140-2 security level 3.

3430 It is difficult of specify quantitative certification test cases for accreditation of these levels. Content
3431 protection regimes usually talk about different tools (widely available, specialized and professional
3432 tools) used to circumvent the hardware protections put in place by manufacturing. If needed, OCF
3433 can follow that model, if and when OCF engage in distributing sensitive key material (e.g. PKI) to
3434 its members.

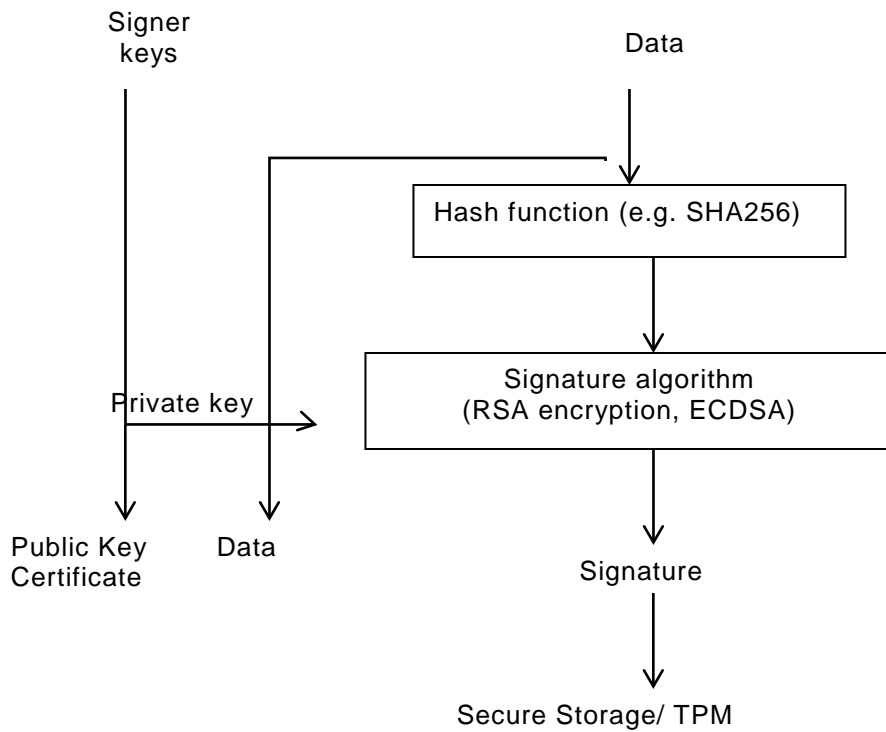
3435 **14.3 Secure Boot**

3436 **14.3.1 Concept of software module authentication**

3437 In order to ensure that all components of a Device are operating properly and have not been
3438 tampered with, it is best to ensure that the Device is booted properly. There may be multiple stages
3439 of boot. The end result is an application running on top an operating system that takes advantage
3440 of memory, CPU and peripherals through drivers.

3441 The general concept is that each software module is invoked only after cryptographic integrity
3442 verification is complete. The integrity verification relies on the software module having been hashed
3443 (e.g. SHA_1, SHA_256) and then signed with a cryptographic signature algorithm with (e.g. RSA),
3444 with a key that only a signing authority has access to.

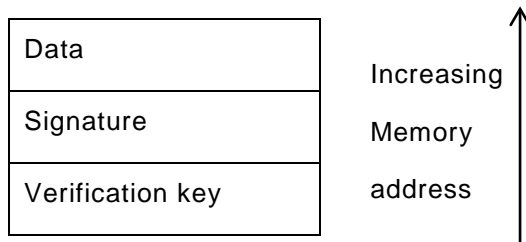
3445 Figure 25 depicts software module authentication.



3446 **Figure 25 – Software Module Authentication**

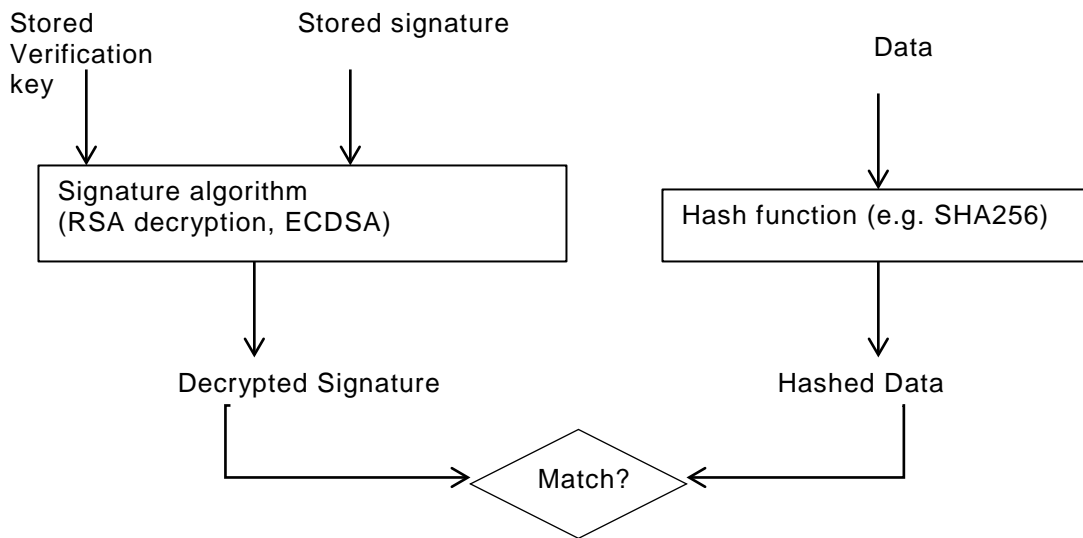
3447 After the data is signed with the signer’s signing key (a private key), the verification key (the public
 3448 key corresponding to the private signing key) is provided for later verification. For lower level
 3449 software modules, such as bootloaders, the signatures and verification keys are inserted inside
 3450 tamper proof memory, such as one-time programmable memory or TPM. For higher level software
 3451 modules, such as application software, the signing is typically performed according to the PKCS#7
 3452 format IETF RFC 2315, where the signedData format includes both indications for signature
 3453 algorithm, hash algorithm as well as the signature verification key (or certificate). Secure boot does
 3454 not require use of PKCS#7 format.

3455 Figure 26 depicts verification software module.



3456 **Figure 26 – Verification Software Module**

3457 As shown in Figure 27. the verification module first decrypts the signature with the verification key
 3458 (public key of the signer). The verification module also calculates a hash of the data and then
 3459 compares the decrypted signature (the original) with the hash of data (actual) and if the two values
 3460 match, the software module is authentic.



3461 **Figure 27 – Software Module Authenticity**

3462 **14.3.2 Secure Boot process**

3463 Depending on the Device implementation, there may be several boot stages. Typically, in a PC/
 3464 Linux type environment, the first step is to find and run the BIOS code (first-stage bootloader) to
 3465 find out where the boot code is and then run the boot code (second-stage boot loader). The second
 3466 stage bootloader is typically the process that loads the operating system (Kernel) and transfers the
 3467 execution to the where the Kernel code is. Once the Kernel starts, it may load external Kernel
 3468 modules and drivers.

3469 When performing a secure boot, it is required that the integrity of each boot loader is verified before
 3470 executing the boot loader stage. As mentioned, while the signature and verification key for the
 3471 lowest level bootloader is typically stored in tamper-proof memory, the signature and verification
 3472 key for higher levels should be embedded (but attached in an easily accessible manner) in the data
 3473 structures software.

3474 **14.3.3 Robustness Requirements**

3475 **14.3.3.1 Robustness General**

3476 To qualify as high robustness secure boot process, the signature and hash algorithms shall be one
 3477 of the approved algorithms, the signature values and the keys used for verification shall be stored
 3478 in secure storage and the algorithms shall run inside a secure execution environment and the keys
 3479 shall be provided the SEE over trusted path.

3480 **14.3.3.2 Next steps**

3481 Develop a list of approved algorithms and data formats

3482 **14.4 Attestation**

3483 **14.5 Software Update**

3484 **14.5.1 Overview:**

3485 The Device lifecycle does not end at the point when a Device is shipped from the manufacturer;
 3486 the distribution, retailing, purchase, installation/onboarding, regular operation, maintenance and
 3487 end-of-life stages for the Device remain outstanding. It is possible for the Device to require update

3488 during any of these stages, although the most likely times are during onboarding, regular operation
 3489 and maintenance. The aspects of the software include, but are not limited to, firmware, operating
 3490 system, networking stack, application code, drivers, etc.

3491 **14.5.2 Recognition of Current Differences**

3492 Different manufacturers approach software update utilizing a collection of tools and strategies:
 3493 over-the-air or wired USB connections, full or partial replacement of existing software, signed and
 3494 verified code, attestation of the delivery package, verification of the source of the code, package
 3495 structures for the software, etc.

3496 It is recommended that manufacturers review their processes and technologies for compliance with
 3497 industry best-practices that a thorough security review of these takes place and that periodic review
 3498 continue after the initial architecture has been established.

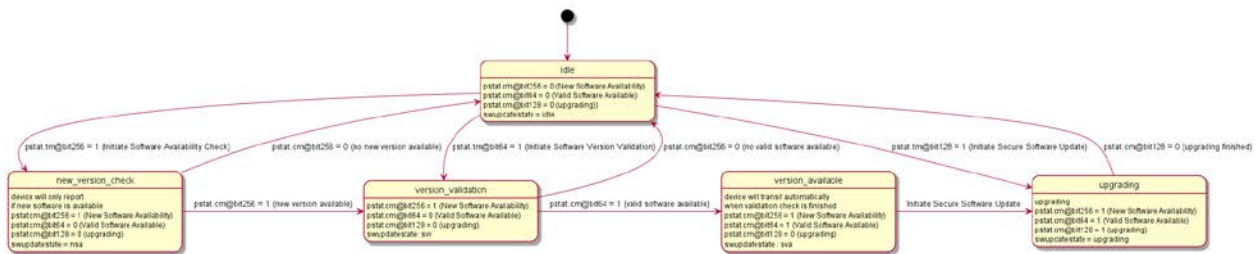
3499 This document applies to software updates as recommended to be implemented by OCF Devices;
 3500 it does not have any bearing on the above-mentioned alternative proprietary software update
 3501 mechanisms. The described steps are being triggered by an OCF Client, the actual implementation
 3502 of the steps and how the software package is downloaded and upgraded is vendor specific.

3503 The triggers that can be invoked from OCF clients can perform:

- 3504 1) Check if new software is available
- 3505 2) Download and verify the integrity of the software package
- 3506 3) Install the verified software package

3507 The triggers are not sequenced, each trigger can be invoked individually.

3508 The state of the transitions of software update is in Figure 28.



3509
3510 **Figure 28 – State transitioning diagram for software download**

3511
3512 **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

3513
3514 **14.5.2.1 Checking availability of new software**
 3515 Setting the Initiate Software Availability Check bit in the "/oic/sec/pstat.tm" Property (see Table 37
 3516 of clause 13.8) indicates a request to initiate the process to check if new software is available, e.g.
 3517 the process whereby the Device checks if a newer software version is available on the external

3518 endpoint. Once the Device has determined if a newer software version is available, it sets the
3519 Initiate Software Availability Check bit in the "/oic/sec/pstat.cm" Property to 1 (TRUE), indicating
3520 that new software is available or to 0 (FALSE) if no newer software version is available, See also
3521 Table 50 where the bits in property TM indicates that the action is initiated and the CM bits are
3522 indicating the result of the action. The Device receiving this trigger is not downloading and not
3523 validating the software to determine if new software is available. The version check is determined
3524 by the current software version and the software version on the external endpoint. The
3525 determination if a software package is newer is vendor defined.

3526 **14.5.3 Software Version Validation**

3527 Setting the Initiate Software Version Validation bit in the "/oic/sec/pstat.tm" Property (see Table 37
3528 of 13.8) indicates a request to initiate the software version validation process, the process whereby
3529 the Device validates the software (including firmware, operating system, Device drivers, networking
3530 stack, etc.) against a trusted source to see if, at the conclusion of the check, the software update
3531 process will need to be triggered (see clause 14.5.4). When the Initiate Software Version Validation
3532 bit of "/oic/sec/pstat.tm" is set to 1 (TRUE) by a sufficiently privileged Client, the Device sets the
3533 "/oic/sec/pstat.cm" Initiate Software Version Validation bit to 0 and initiates a software version
3534 check. Once the Device has determined if a valid software is available, it sets the Initiate Software
3535 Version Validation bit in the "/oic/sec/pstat.cm" Property to 1 (TRUE) if an update is available or 0
3536 (FALSE) if no update is available. To signal completion of the Software Version Validation process,
3537 the Device sets the Initiate Software Version Validation bit in the "/oic/sec/pstat.tm" Property back
3538 to 0 (FALSE). If the Initiate Software Version Validation bit of "/oic/sec/pstat.tm" is set to 0 (FALSE)
3539 by a Client, it has no effect on the validation process. The Software Version Validation process can
3540 download the software from the external endpoint to verify the integrity of the software package.

3541 **14.5.4 Software Update**

3542 Setting the Initiate Secure Software Update bit in the "/oic/sec/pstat.tm" Property (see Table 37 of
3543 clause 13.8) indicates a request to initiate the software update process. When the Initiate Secure
3544 Software Update bit of "/oic/sec/pstat.tm" is set to 1 (TRUE) by a sufficiently privileged Client, the
3545 Device sets the "/oic/sec/pstat.cm" Initiate Software Version Validation bit to 0 and initiates a
3546 software update process. Once the Device has completed the software update process, it sets the
3547 Initiate Secure Software Update bit in the "/oic/sec/pstat.cm" Property to 1 (TRUE) if/when the
3548 software was successfully updated or 0 (FALSE) if no update was performed. To signal completion
3549 of the Secure Software Update process, the Device sets the Initiate Secure Software Update bit in
3550 the "/oic/sec/pstat.tm" Property back to 0 (FALSE). If the Initiate Secure Software Update bit of
3551 "/oic/sec/pstat.tm" is set to 0 (FALSE) by a Client, it has no effect on the update process.

3552 **14.5.4.1 State of Device after software update**

3553 The state of all resources implemented in the Device should be the same as after boot, meaning
3554 that the software update is not resetting user data and retaining a correct state.

3555 User data of a Device is defined as:

- 3556 – Retain the SVR states, e.g. the on boarded state, registered clients.
- 3557 – Retain all created resources
- 3558 – Retain all stored data of a resource
- 3559 – For example the preferences stored for the brewing resource ("oic.r.brewing").

3560 **14.5.5 Recommended Usage**

3561 The Initiate Secure Software Update bit of "/oic/sec/pstat.tm" should only be set by a Client after
3562 the Initiate Software Version Validation check is complete.

3563 The process of updating Device software may involve state changes that affect the Device
3564 Operational State ("/oic/sec/pstat.dos"). Devices with an interest in the Device(s) being updated

3565 should monitor "/oic/sec/pstat.dos" and be prepared for pending software update(s) to affect Device
3566 state(s) prior to completion of the update.

3567 The Device itself may indicate that it is autonomously initiating a software version check/update or
3568 that a check/update is complete by setting the "pstat.tm" and "pstat.cm" Initiate Software Version
3569 Validation and Secure Software Update bits when starting or completing the version check or
3570 update process. As is the case with a Client-initiated update, Clients can be notified that an
3571 autonomous version check or software update is pending and/or complete by observing pstat
3572 resource changes.

3573 The "oic.r.softwareupdate" Resource Type specifies additional features to control the software
3574 update process see core specification.

3575 **14.6 Non-OCF Endpoint interoperability**

3576 **14.7 Security Levels**

3577 Security Levels are a way to differentiate Devices based on their security criteria. This need for
3578 differentiation is based on the requirements from different verticals such as industrial and health
3579 care and may extend into smart home. This differentiation is distinct from Device classification
3580 (e.g. IETF RFC 7228)

3581 These categories of security differentiation may include, but is not limited to:

- 3582 1) Security Hardening
- 3583 2) Identity Attestation
- 3584 3) Certificate/Trust
- 3585 4) Onboarding Technique
- 3586 5) Regulatory Compliance
 - 3587 a) Data at rest
 - 3588 b) Data in transit
- 3589 6) Cipher Suites – Crypto Algorithms & Curves
- 3590 7) Key Length
- 3591 8) Secure Boot/Update

3592 In the future security levels can be used to define interoperability.

3593 The following applies to the OCF Security Specification 1.1:

3594 The current document does not define any other level beyond Security Level 0. All Devices will be
3595 designated as Level 0. Future versions may define additional levels.

3596 Additional comments:

- 3597 – The definition of a given security level will remain unchanged between versions of the document.
- 3598 – Devices that meet a given level may, or may not, be capable of upgrading to a higher level.
- 3599 – Devices may be evaluated and re-classified at a higher level if it meets the requirements of the
3600 higher level (e.g. if a Device is manufactured under the 1.1 version of the document, and a later
3601 document version defines a security level 1, the Device could be evaluated and classified as
3602 level 1 if it meets level 1 requirements).
- 3603 – The security levels may need to be visible to the end user.

3604 **14.8 Security Profiles**

3605 **14.8.1 Security Profiles General**

3606 Security Profiles are a way to differentiate OCF Devices based on their security criteria. This need
3607 for differentiation is based on the requirements from different verticals such as industrial and health
3608 care and may extend into smart home. This differentiation is distinct from device classification (e.g.
3609 IETF RFC 7228)

3610 These categories of security differentiation may include, but is not limited to:

- 3611 1) Security Hardening and assurances criteria
- 3612 2) Identity Attestation
- 3613 3) Certificate/Trust
- 3614 4) Onboarding Technique
- 3615 5) Regulatory Compliance
 - 3616 a) Data at rest
 - 3617 b) Data in transit
- 3618 6) Cipher Suites – Crypto Algorithms & Curves
- 3619 7) Key Length
- 3620 8) Secure Boot/Update

3621 Each Security Profile definition must specify the version or versions of the OCF Security
3622 Specification(s) that form a baseline set of normative requirements. The profile definition may
3623 include security requirements that supersede baseline requirements (not to relax security
3624 requirements).

3625 Security Profiles have the following properties:

- 3626 – A given profile definition is not specific to the version of the document that defines it. For
3627 example, the profile may remain constant for subsequent OCF Security Specification versions.
- 3628 – A specific OCF Device and platform combination may be used to satisfy the security profile.
- 3629 – Profiles may have overlapping criteria; hence it may be possible to satisfy multiple profiles
3630 simultaneously.
- 3631 – An OCF Device that satisfied a profile initially may be re-evaluated at a later time and found to
3632 satisfy a different profile (e.g. if a device is manufactured under the 1.1 version of the document,
3633 and a later document version defines a security profile Black, the device could be evaluated
3634 and classified as profile Black if it meets profile Black requirements).
- 3635 – A machine-readable representation of compliance results specifically describing profiles
3636 satisfied may be used to facilitate OCF Device onboarding. (e.g. a manufacturer certificate or
3637 manifest may contain security profiles attributes).

3638 **14.8.2 Identification of Security Profiles (Normative)**

3639 **14.8.2.1 Security Profiles in Prior Documents**

3640 OCF Devices conforming to versions of the OCF Security Specifications where Security Profiles
3641 Resource was not defined may be presumed to satisfy the "sp-baseline-v0" profile (defined in
3642 14.8.3.3) or may be regarded as unspecified. If Security Profile is unspecified, the Client may use
3643 the OCF Security Specification version to characterize expected security behaviour.

3644 **14.8.2.2 Security Profile Resource Definition**

3645 The "/oic/sec/sp" Resource is used by the OCF Device to show which OCF Security Profiles the
3646 OCF Device is capable of supporting and which are authorized for use by the OCF Security Domain

3647 owner. Properties of the Resource identify which OCF Security Profile is currently operational. The
 3648 ocfSecurityProfileOID value type shall represent OID values and may reference an entry in the form
 3649 of strings (UTF-8).

3650 "/oic/sec/sp" Resource is defined in Table 51.

3651 **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

3652 Table 52 defines the Properties of "/oic/sec/sp" Resource.

3653 **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")

3654 The following OIDs are defined to uniquely identify Security Profiles. Future Security Profiles or
 3655 changes to existing Security Profiles may result in a new ocfSecurityProfileOID.

3656 id-OCF OBJECT IDENTIFIER ::= { iso(1) identified-organization(3) dod(6) internet(1)
 3657 private(4) enterprise(1) OCF(51414) }

3658
 3659 id-ocfSecurity OBJECT IDENTIFIER ::= { id-OCF 0 }

3660
 3661 id-ocfSecurityProfile ::= { id-ocfSecurity 0 }

3662
 3663 sp-undefined ::= OBJECT IDENTIFIER { id-ocfSecurityProfile 0 }

3664 --The Security Profile is not specified

3665 sp-baseline ::= OBJECT IDENTIFIER { id-ocfSecurityProfile 1 }

3666 --This specifies the OCF Baseline Security Profile(s)

3667 sp-black ::= OBJECT IDENTIFIER { id-ocfSecurityProfile 2 }

3668 --This specifies the OCF Black Security Profile(s)

3669 sp-blue ::= OBJECT IDENTIFIER { id-ocfSecurityProfile 3 }

3670 --This specified the OCF Blue Security Profile(s)

3671 sp-purple ::= OBJECT IDENTIFIER { id-ocfSecurityProfile 4 }

3672 --This specifies the OCF Purple Security Profile(s)

3673
 3674 --versioned Security Profiles

3675 sp-undefined-v0 ::= ocfSecurityProfileOID (id-sp-undefined 0)

3676 --v0 of undefined security profile, "1.3.6.1.4.1.51414.0.0.0.0"

3677 sp-baseline-v0 ::= ocfSecurityProfileOID {id-sp-baseline 0}

3678 --v0 of baseline security profile, "1.3.6.1.4.1.51414.0.0.1.0"

3679 sp-black-v0 ::= ocfSecurityProfileOID {id-sp-black 0}

3680 --v0 of black security profile, "1.3.6.1.4.1.51414.0.0.2.0"

3681 sp-blue-v0 ::= ocfSecurityProfileOID {id-sp-blue 0}

3682 --v0 of blue security profile, "1.3.6.1.4.1.51414.0.0.3.0"

3683 sp-purple-v0 ::= ocfSecurityProfileOID {id-sp-purple 0}

3684 --v0 of purple security profile, "1.3.6.1.4.1.51414.0.0.4.0"

3685
 3686 ocfSecurityProfileOID ::= UTF8String

3687

3688 **14.8.3 Security Profiles**

3689 **14.8.3.1 Security Profiles General**

3690 The Security Profiles Resource shall be pre-populated with manufacturer default values (Refer to
3691 the Security Profile clauses for additional details).

3692 The OCF Conformance criteria may require vendor attestation that establishes the expected
3693 environment in which the OCF Device is hosted (Refer to the Security Profile clauses for specific
3694 requirements).

3695 **14.8.3.2 Security Profile Unspecified (sp-unspecified-v0)**

3696 The Security Profile "sp-unspecified-v0" is reserved for future use.

3697 **14.8.3.3 Security Profile Baseline v0 (sp-baseline-v0)**

3698 The Security Profile "sp-baseline-v0" is defined for all OCF Security Specification versions where
3699 the "/oic/sec/sp" Resource is defined. All Devices shall include the "sp-baseline-v0" OID in the
3700 "supportedprofiles" Property of the "/oic/sec/sp" Resource.

3701 It indicates the OCF Device satisfies the normative security requirements for this document.

3702 When a device supports the baseline profile, the "supportedprofiles" Property shall contain sp-
3703 baseline-v0, represented by the OID string "1.3.6.1.4.1.51414.0.0.1.0", and may contain other
3704 profiles.

3705 When a manufacturer makes sp-baseline-v0 the default, by setting the "currentprofile" Property to
3706 "1.3.6.1.4.1.51414.0.0.1.0", the "supportedprofiles" Property shall contain sp-baseline-v0.

3707 **14.8.3.4 Security Profile Black (sp-black-v0)**

3708 **14.8.3.4.1 Black Profile General**

3709 The need for Security Profile Black v0 is to support devices and manufacturers who wish to certify
3710 their devices meeting this specific set of security criteria. A Device may satisfy the Black
3711 requirements as well as requirements of other profiles, the Black Security Profile is not necessarily
3712 mutually exclusive with other Security Profiles unless those requirements conflict with the explicit
3713 requirements of the Black Security Profile.

3714 **14.8.3.4.2 Devices Targeted for Security Profile Black v0**

3715 Security Profile Black devices could include any device a manufacturer wishes to certify at this
3716 profile, but healthcare devices and industrial devices with additional security requirements are the
3717 initial target. Additionally, manufacturers of devices at the edge of the network (or fog), or devices
3718 with exceptional profiles of trust bestowed upon them, may wish to certify at this profile; these types
3719 of devices may include, but are not limited to the following:

- 3720 – Bridges (Mapping devices between ecosystems handling virtual devices from different
3721 ecosystems)
- 3722 – Resource Directories (Devices trusted to manage OCF Security Domain resources)
- 3723 – Remote Access (Devices which have external access but can also act within the OCF Security
3724 Domain)
- 3725 – Healthcare Devices (Devices with specific requirements for enhanced security and privacy)
- 3726 – Industrial Devices (Devices with advanced management, security and attestation requirements)

3727 **14.8.3.4.3 Requirements for Certification at Security Profile Black (Normative)**

3728 Every device with "currentprofile" Property of the "/oic/sec/sp" Resource designating a Security
3729 Profile of "sp-black-v0", as defined in clause 14.8.2, must support each of the following:

- 3730 – Onboarding via OCF Rooted Certificate Chain, including PKI chain validation
- 3731 – Support for AES 128 encryption for data at rest and in transit.
- 3732 – Hardening minimums: manufacturer assertion of secure credential storage
- 3733 – In – in enumerated item #10 "The "/oic/sec/cred" Resource should contain credential(s) if
3734 required by the selected OTM" is changed to require the credential be stored: "The
3735 "/oic/sec/cred" Resource shall contain credential(s)."
- 3736 – The OCF Device shall include an X.509v3 OCF Compliance Extension (clause 9.4.2.2.4) in its
3737 certificate and the extension's 'securityProfile' field shall contain sp-black-v0 represented by
3738 the ocfSecurityProfileOID string, "1.3.6.1.4.1.51414.0.0.2.0".

3739 When a device supports the black profile, the "supportedprofiles" Property shall contain sp-black-
3740 v0, represented by the OID string "1.3.6.1.4.1.51414.0.0.2.0", and may contain other profiles.

3741 When a manufacturer makes sp-black-v0 the default, by setting the "currentprofile" Property to
3742 "1.3.6.1.4.1.51414.0.0.2.0", the "supportedprofiles" Property shall contain sp-black-v0.

3743 The OCF Rooted Certificate Chain and PKI Is defined by and structured within a framework
3744 described in the supporting documents:

- 3745 – Certificate Profile (See 9.4.2)
- 3746 – Certificate Policy (see Certificate Policy document:
3747 <https://openconnectivity.org/specs/OCF%20Certificate%20Policy.pdf>)

3748 **14.8.3.5 Security Profile Blue v0 (sp-blue-v0)**

3749 **14.8.3.5.1 Blue Profile General**

3750 The Security Profile Blue is used when manufacturers issue platform certificates for platforms
3751 containing manufacturer-embedded keys. Compatibility with interoperable trusted platforms is
3752 anticipated using certificate extensions defined by the Trusted Computing Group (TCG). OCF
3753 Security Domain owners evaluate manufacturer supplied certificates and attributed data to
3754 determine an appropriate OCF Security Profile that is configured for OCF Devices at onboarding.
3755 OCF Devices may satisfy multiple OCF Security Profiles. The OCF Security Domain owner may
3756 configure deployments using the Security Profile as OCF Security Domain partitioning criteria.

3757 Certificates issued to Blue Profile Devices shall be issued by a CA conforming to the CA Vetting
3758 Criteria defined by OCF.

3759 **14.8.3.5.2 Platforms and Devices for Security Profile Blue v0**

3760 The OCF Security Profile Blue anticipates an ecosystem where platform vendors may differ from
3761 OCF Device vendor and where platform vendors may implement trusted platforms that may conform
3762 to industry standards defining trusted platforms. The OCF Security Profile Blue specifies
3763 mechanisms for linking platforms with OCF Device(s) and for referencing quality assurance criteria
3764 produced by OCF conformance operations. The OCF Security Domain owner evaluates these data
3765 when an OCF Device is onboarded into the OCF Security Domain. Based on this evaluation the
3766 OCF Security Domain owner determines which Security Profile may be applied during OCF Device
3767 operation. All OCF Device types may be considered for evaluation using the OCF Security Profile
3768 Blue.

3769 **14.8.3.5.3 Requirements for Certification at Security Profile Blue v0**

3770 The OCF Device satisfies the Blue profile v0 (sp-blue-v0) when all of the security normative for this
3771 document version are satisfied and the following additional criteria are satisfied.

3772 OCF Blue profile defines the following OCF Device quality assurances:

- 3773 – The OCF Conformance criteria shall require vendor attestation that the conformant OCF Device
3774 was hosted on one or more platforms that satisfies OCF Blue platform security assurances and
3775 platform security and privacy functionality requirements.
- 3776 – The OCF Device achieving OCF Blue Security Profile compliance will be registered by OCF and
3777 published by OCF in a machine readable format.
- 3778 – The OCF Blue Security Profile compliance registry may be digitally signed by an OCF owned
3779 signing key.
- 3780 – The OCF Device shall include an X.509v3 OCF Compliance Extension (clause 9.4.2.2.4) in its
3781 certificate and the extension's 'securityProfile' field shall contain sp-blue-v0 represented by the
3782 ocfSecurityProfileOID string, "1.3.6.1.4.1.51414.0.0.3.0".
- 3783 – The OCF Device shall include an X.509v3 OCF CPL Attributes Extension (clause 9.4.2.2.7) in
3784 its certificate.
- 3785 – The DOTS is expected to perform a lookup of the certification status of the OCF Device using
3786 the OCF CPL Attributes Extension values and verify that the sp-blue-v0 OID is listed in the
3787 extension's "securityprofiles" field.

3788 OCF Blue profile defines the following OCF Device security functionality:

- 3789 – OCF Device(s) shall be hosted on a platform where a cryptographic and secure storage
3790 functions are hardened by the platform.
- 3791 – OCF Device(s) hosted on a platform shall expose accompanying manufacturer credentials using
3792 the "/oic/sec/cred" Resource where the "credusage" Property contains the value
3793 "oic.sec.cred.mfgcert".
- 3794 – OCF Device(s) that are hosted on a TCG-defined trusted platform should use an IEEE802.1AR
3795 IDevID and should verify the "TCG Endorsement Key Credential". All TCG-defined
3796 manufacturer credentials may be identified by the "oic.sec.cred.mfgcert" value of the
3797 "credusage" Property of the "/oic/sec/cred" Resource. They may be used in response to
3798 selection of the "oic.sec.doxm.mfgcert" owner transfer method.
- 3799 – OCF Device(s) shall use AES128 equivalent minimum protection for transmitted data. (See
3800 NIST SP 800-57).
- 3801 – OCF Device(s) shall use AES128 equivalent minimum protection for stored data. (See NIST SP
3802 800-57).
- 3803 – OCF Device(s) should use AES256 equivalent minimum protection for stored data. (See NIST
3804 SP 800-57).
- 3805 – OCF Device(s) should protect the "/oic/sec/cred" resource using the platform provided secure
3806 storage.
- 3807 – OCF Device(s) shall protect trust anchors (aka policy defining trusted CAs and pinned
3808 certificates) using platform provided secure storage.
- 3809 – OCF Device(s) should check certificate revocation status for locally issued certificates.
- 3810 – The DOTS is expected to check certificate revocation status for all certificates in manufacturer
3811 certificate path(s) if available. If a certificate is revoked, certificate validation fails and the
3812 connection is refused. The DOTS may disregard revocation status results if unavailable.

3813 OCF Blue profile defines the following platform security assurances:

- 3814 – Platforms implementing cryptographic service provider (CSP) functionality and secure storage
3815 functionality should be evaluated with a minimum FIPS140-2 Level 2 or Common Criteria EAL
3816 Level 2.

- 3817 – Platforms implementing trusted platform functionality should be evaluated with a minimum
3818 Common Criteria EAL Level 1.
- 3819 OCF Blue profile defines the following platform security and privacy functionality:
- 3820 – The Platform shall implement cryptographic service provider (CSP) functionality.
 - 3821 – Platform CSP functionality shall include cryptographic algorithms, random number generation,
3822 secure time.
 - 3823 – The Platform shall implement AES128 equivalent protection for transmitted data. (See NIST SP
3824 800-57).
 - 3825 – The Platform shall implement AES128 and AES256 equivalent protection for stored data. (See
3826 NIST SP 800-57).
 - 3827 – Platforms hosting OCF Device(s) should implement a platform identifier following IEEE802.1AR
3828 or Trusted Computing Group(TCG) specifications.
 - 3829 – Platforms based on Trusted Computing Group (TCG) platform definition that host OCF Device(s)
3830 should supply TCG-defined manufacture certificates; also known as "TCG Endorsement Key
3831 Credential" (which complies with IETF RFC 5280) and "TCG Platform Credential" (which
3832 complies with IETF RFC 5755).
- 3833 When a device supports the blue profile, the "supportedprofiles" Property shall contain sp-blue-v0,
3834 represented by the OID string "1.3.6.1.4.1.51414.0.0.3.0", and may contain other profiles.
- 3835 When a manufacturer makes sp-blue-v0 the default, by setting the "currentprofile" Property to
3836 "1.3.6.1.4.1.51414.0.0.3.0", the "supportedprofiles" Property shall contain sp-blue-v0.
- 3837 During onboarding, while the device state is RFOTM, the DOTS may update the "currentprofile"
3838 Property to one of the other values found in the "supportedprofiles" Property.
- 3839 **14.8.3.6 Security Profile Purple v0 (sp-purple-v0)**
- 3840 Every device with the "/oic/sec/sp" Resource designating "sp-purple-v0", as defined in clause
3841 14.8.2 must support following minimum requirements
- 3842 – Hardening minimums: secure credential storage, software integrity validation, secure update.
 - 3843 – If a Certificate is used, the OCF Device shall include an X.509v3 OCF Compliance Extension
3844 (clause 9.4.2.2.4) in its certificate and the extension's 'securityProfile' field shall contain sp-
3845 purple-v0 represented by the ocfSecurityProfileOID string, "1.3.6.1.4.1.51414.0.0.4.0"
 - 3846 – The OCF Device shall include a X.509v3 OCFPLAttributes Extension (clause 9.4.2.2.7) in its
3847 End-Entity Certificate when manufacturer certificate is used.
- 3848 Security Profile Purple has following optional security hardening requirements that the device can
3849 additionally support.
- 3850 – Hardening additions: secure boot, hardware backed secure storage
 - 3851 – The OCF Device shall include a X.509v3 OCFSecurityClaims Extension (clause 9.4.2.2.6) in its
3852 End-Entity Certificate and it shall include corresponding OIDs to the hardening additions
3853 implemented and attested by the vendor. If there is no additional support for hardening
3854 requirements, X.509v3 OCFSecurityClaims Extension shall be omitted.
- 3855 For software integrity validation, OCF Device(s) shall provide the integrity validation mechanism
3856 for security critical executables such as cryptographic modules or secure service applications, and
3857 they should be validated before the execution. The key used for validating the integrity must be
3858 pinned at the least to the validating software module.
- 3859 For secure update, OCF Device(s) shall be able to update its firmware in a secure manner.

3860 For secure boot, OCF Device(s) shall implement the BIOS code (first-stage bootloader on ROM) to
3861 be executed by the processor on power-on, and secure boot parameters to be provisioned by
3862 tamper-proof memory. Also OCF Device(s) shall provide software module authentication for the
3863 security critical executables and stop the boot process if any integrity of them is compromised.

3864 For hardware backed secure storage, OCF Device(s) shall store sensitive data in non-volatile
3865 memory ("NVRAM") and prevent the retrieval of sensitive data through physical and/or electronic
3866 attacks.

3867 More details on security hardening guidelines for software integrity validation, secure boot, secure
3868 update, and hardware backed secure storage are described in 14.3, 14.5 and 14.2.2.2.

3869 Certificates issued to Purple Profile Devices shall be issued by a CA conforming to the CA Vetting
3870 Criteria defined by OCF.

3871 When a device supports the purple profile, the "supportedprofiles" Property shall contain sp-purple-
3872 v0, represented by the OID string "1.3.6.1.4.1.51414.0.0.4.0", and may contain other profiles.

3873 When a manufacturer makes sp-purple-v0 the default, by setting the "currentprofile" Property to
3874 "1.3.6.1.4.1.51414.0.0.4.0", the "supportedprofiles" Property shall contain sp-purple-v0.

3875 **15 Device Type Specific Requirements**

3876 **15.1 Bridging Security**

3877 **15.1.1 Universal Requirements for Bridging to another Ecosystem**

3878 The Bridge shall go through OCF ownership transfer as any other onboarder would.

3879 The software of a Bridge shall be field updatable. (This requirement need not be tested but can be
3880 certified via a vendor declaration.)

3881 Each VOD shall be onboarded by an OCF OBT. Each Virtual Bridged Device should be provisioned
3882 as appropriate in the Bridged Protocol. In other words, VODs and Virtual Bridged Devices are
3883 treated the same way as physical Devices. They are entities that have to be provisioned in their
3884 network.

3885 Each VOD shall implement the behaviour required by ISO/IEC 30118-1:2018 and this document.
3886 Each VOD shall perform authentication, access control, and encryption according to the security
3887 settings it received from the OCF OBT. Each Virtual Bridged Device shall implement the security
3888 requirements of the Bridged Protocol.

3889 In addition, in order to be considered secure from an OCF perspective, the Bridge Platform shall
3890 use appropriate ecosystem-specific security options for communication between the Virtual Bridged
3891 Devices instantiated by the Bridge and Bridged Devices. This security shall include mutual
3892 authentication, and encryption and integrity protection of messages in the bridged ecosystem.

3893 A VOD may authenticate itself to the DOTS using the Manufacturer Certificate Based OTM (see
3894 clause 7.3.6) with the Manufacturer Certificate and corresponding private key of the Bridge which
3895 instantiated that VOD.

3896 A VOD may authenticate itself to the OCF Cloud using the Manufacturer Certificate and
3897 corresponding private key of the Bridge which instantiated that VOD.

3898 A Bridge and the VODs created by that Bridge shall operate as independent Devices, with the
3899 following exceptions:

3900 – If a Bridge creates a VOD while the Bridge is in an Unowned State, then the VOD shall be
3901 created in an Unowned State.

3902 – An Unowned VOD shall not accept DTLS connection attempts nor TLS connection attempts nor
3903 any other requests, including discovery requests, while the Bridge (that created that VOD) is
3904 Unowned.

3905 – At any time when a Bridge is transitioning from Owned to Unowned State, all Unowned VODs
3906 (created by that Bridge prior to the transition) shall drop any existing TLS and/or DTLS
3907 connections.

3908 – At any time when a Bridge is transitioning from Unowned to Owned State, the Bridge shall
3909 trigger all Unowned VODs (created by that Bridge prior to the transition) to become accessible
3910 in RFOTM state, with internal state as if the VOD has just transitioned from RESET to RFOTM.

3911 – If a Bridge creates a VOD while the Bridge is in an Owned State, then the VOD shall become
3912 accessible in RFOTM state, with internal state as if the VOD has just transitioned from RESET
3913 to RFOTM.

3914 Table 53 intends to clarify this behaviour.

3915
3916

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

3917 The "vods" Property of the "oic.r.vodlist" Resource on a Bridge reflects the details of all currently
 3918 Owned VODs which have been created by that Bridge since the most recent hardware reset (if any)
 3919 of the Bridge Platform (which removes all the created VODs), regardless of whether the VODs have
 3920 the same owner as the Bridge or not. The entries in the "vods" Property are added and removed
 3921 according to the following criteria:

- 3922 – Whenever a VOD created by a Bridge transitions from being Unowned to being Owned, then
 3923 an entry for that VOD shall be added to the "vods" Property of the "oic.r.vodlist" Resource of
 3924 that Bridge.
- 3925 – Whenever a VOD created by a Bridge transitions from being Owned to being Unowned, then
 3926 entry for that VOD shall be removed from the "vods" Property of the "oic.r.vodlist" Resource of
 3927 that Bridge. If that Bridge is currently in Unowned state, then the "oic.r.vodlist" Resource is not
 3928 accessible, and the entry for that VOD shall be removed from the "vods" Property before or
 3929 during the transition of that Bridge to the Owned state.
- 3930 – All other modifications of the list are not allowed.

3931 A Bridge shall only expose a secure OCF Endpoint for the "oic.r.vodlist" Resource.

3932 **15.1.2 Additional Security Requirements specific to Bridged Protocols**

3933 **15.1.2.1 Additional Security Requirements specific to the AllJoyn Protocol**

3934 For AllJoyn translator, an authenticated and authorized Client shall be able to block the
 3935 communication of all OCF Devices with all Bridged Devices that don't communicate securely with
 3936 the Bridge, by using the Bridge Device's "oic.r.securemode" Resource specified in ISO/IEC 30118-
 3937 3:2018

3938 **15.1.2.2 Additional Security Requirements specific to the Bluetooth LE 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 **15.1.2.3 Additional Security Requirements specific to the oneM2M Protocols**

3942 The Bridge shall implement oneM2M application access control as defined in the oneM2M Release
 3943 3 Specifications.

3944 An Bridge shall block the communication of all OCF Devices with all Bridged Devices that don't
 3945 communicate securely with the Bridge.

3946 **15.1.2.4 Additional Security Requirements specific to the U+ Protocol**

3947 A Bridge shall block the communication of all OCF Devices with all Bridged Devices that don't
3948 communicate securely with the Bridge.

3949 **15.1.2.5 Additional Security Requirements specific to the Z-Wave Protocol**

3950 An Bridge shall block the communication of all OCF Devices with all Bridged Devices that don't
3951 communicate securely with the Bridge.

3952 **15.1.2.6 Additional Security Requirements specific to the Zigbee Protocol**

3953 An Bridge shall block the communication of all OCF Devices with all Bridged Devices that don't
3954 communicate securely with the Bridge.

3955

3956

3957

3958

3959

3960

3961

3962

3963

3964

3965

3966

3967

3968

3969

3970

3971

3972

3973

3974

3975

3976 .

Annex A
(informative)
Access Control Examples

3977
3978
3979

Example OCF ACL Resource

3980

3981 Figure A-1 shows how a "/oic/sec/acl2" Resource could be configured to enforce an example
3982 access policy on the Server.

```
3983 {
3984   "aclist2": [
3985     {
3986       // Subject with ID ...01 should access two named Resources with access mode "CRUDN" (Create, Retrieve, Update,
3987       Delete and Notify)
3988       "subject": {"uuid": "XXXX-...-XX01"},
3989       "resources": [
3990         {"href": "/oic/sh/light/1"},
3991         {"href": "/oic/sh/temp/0"}
3992       ],
3993       "permission": 31, // 31 dec = 0b0001 1111 which maps to ---N DURC
3994       "validity": [
3995         // The period starting at 18:00:00 UTC, on January 1, 2015 and
3996         // ending at 07:00:00 UTC on January 2, 2015
3997         "period": ["20150101T180000Z/20150102T070000Z"],
3998         // Repeats the {period} every week until the last day of Jan. 2015.
3999         "recurrence": ["RRULE:FREQ=WEEKLY;UNTIL=20150131T070000Z"]
4000       ],
4001       "aceid": 1
4002     }
4003   ],
4004   // An ACL provisioning and management service should be identified as
4005   // the resource owner
4006   "rowneruuid": "0685B960-736F-46F7-BEC0-9E6CBD61ADC1"
4007 }
4008
```

Figure A-1 – Example "/oic/sec/acl2" Resource

4009
4010
4011

Annex B (Informative) Execution Environment Security Profiles

4012 Given that IoT verticals and Devices will not be of uniform capabilities, a one-size-fits all security
4013 robustness requirements meeting all IOT applications and services will not serve the needs of OCF,
4014 and security profiles of varying degree of robustness (trustworthiness), cost and complexity have
4015 to be defined. To address a large ecosystem of vendors, the profiles can only be defined as
4016 requirements and the exact solutions meeting those requirements are specific to the vendors' open
4017 or proprietary implementations, and thus in most part outside scope of this document.

4018 To align with the rest of OCF documents, where Device classifications follow IETF RFC 7228
4019 (Terminology for constrained node networks) methodology, we limit the number of security profiles
4020 to a maximum of 3 (see Table B.1). However, our understanding is OCF capabilities criteria for
4021 each of 3 classes will be more fit to the current IoT chip market than that of IETF.

4022 Given the extremely low level of resources at class 0, our expectation is that class 0 Devices are
4023 either capable of no security functionality or easily breakable security that depend on environmental
4024 (e.g. availability of human) factors to perform security functions. This means the class 0 will not be
4025 equipped with an SEE.

4026

Table B.1 – OCF Security Profile

Platform class	SEE	Robustness level
0	No	N/A
1	Yes	Low
2	Yes	High

4027 NOTE This analysis acknowledges that these Platform classifications do not take into consideration of possibility of
4028 security co-processor or other hardware security capability that augments classification criteria (namely CPU speed,
4029 memory, storage).

4030
4031
4032

Annex C (normative) Resource Type definitions

4033 C.1 List of Resource Type definitions

4034 Table C.1 contains the list of defined security resources in this document.

4035 **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

4036 C.2 Access Control List-2

4037 C.2.1 Introduction

4038 This Resource specifies the local access control list.
4039 When used without query parameters, all the ACE entries are returned.
4040 When used with a query parameter, only the ACEs matching the specified
4041 parameter are returned.
4042

4043 C.2.2 Well-known URI

4044 /oic/sec/acl2

4045 C.2.3 Resource type

4046 The Resource Type is defined as: "oic.r.acl2".

4047 C.2.4 OpenAPI 2.0 definition

```
4048 {  
4049   "swagger": "2.0",  
4050   "info": {  
4051     "title": "Access Control List-2",  
4052     "version": "20190111",  
4053     "license": {  
4054       "name": "OCF Data Model License",  
4055       "url":  
4056       "https://github.com/openconnectivityfoundation/core/blob/e28a9e0a92e17042ba3e83661e4c0fbce8bdc4ba/LI  
4057       CENSE.md",  
4058       "x-copyright": "copyright 2016-2017, 2019 Open Connectivity Foundation, Inc. All rights  
4059       reserved."  
4060     }  
4061   }  
4062 }
```

```

4061     "termsOfService": "https://openconnectivityfoundation.github.io/core/DISCLAIMER.md"
4062   },
4063   "schemes": ["http"],
4064   "consumes": ["application/json"],
4065   "produces": ["application/json"],
4066   "paths": {
4067     "/oic/sec/acl2" : {
4068       "get": {
4069         "description": "This Resource specifies the local access control list.\nWhen used without
4070 query parameters, all the ACE entries are returned.\nWhen used with a query parameter, only the ACEs
4071 matching the specified\nparameter are returned.\n",
4072         "parameters": [
4073           {"$ref": "#/parameters/interface"},
4074           {"$ref": "#/parameters/ace-filtered"}
4075         ],
4076         "responses": {
4077           "200": {
4078             "description": "",
4079             "x-example":
4080               {
4081                 "rt" : ["oic.r.acl2"],
4082                 "aclist2": [
4083                   {
4084                     "aceid": 1,
4085                     "subject": {
4086                       "authority": "484b8a51-cb23-46c0-a5f1-b4aebef50ebe",
4087                       "role": "SOME_STRING"
4088                     },
4089                     "resources": [
4090                       {
4091                         "href": "/light"
4092                       },
4093                       {
4094                         "href": "/door"
4095                       }
4096                     ],
4097                     "permission": 24
4098                   },
4099                   {
4100                     "aceid": 2,
4101                     "subject": {
4102                       "uuid": "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9"
4103                     },
4104                     "resources": [
4105                       {
4106                         "href": "/light"
4107                       },
4108                       {
4109                         "href": "/door"
4110                       }
4111                     ],
4112                     "permission": 24
4113                   },
4114                   {
4115                     "aceid": 3,
4116                     "subject": {"conntype": "anon-clear"},
4117                     "resources": [
4118                       {
4119                         "href": "/light"
4120                       },
4121                       {
4122                         "href": "/door"
4123                       }
4124                     ],
4125                     "permission": 16,
4126                     "validity": [
4127                       {
4128                         "period": "20160101T180000Z/20170102T070000Z",
4129                         "recurrence": [ "DSTART:XXXXX",
4130 "RRULE:FREQ=DAILY;UNTIL=20180131T140000Z;BYMONTH=1" ]
4131                       },

```

```

4132         {
4133             "period": "20160101T180000Z/PT5H30M",
4134             "recurrence": [ "RRULE:FREQ=DAILY;UNTIL=20180131T140000Z;BYMONTH=1" ]
4135         }
4136     ]
4137 }
4138 ],
4139     "rowneruuid": "de305d54-75b4-431b-adb2-eb6b9e546014"
4140 },
4141     "schema": { "$ref": "#/definitions/Acl2" }
4142 },
4143     "400": {
4144         "description": "The request is invalid."
4145     }
4146 }
4147 },
4148     "post": {
4149         "description": "Updates the ACL Resource with the provided ACEs.\n\nACEs provided in the
4150 update with aceids not currently in the ACL\nResource are added.\n\nACEs provided in the update with
4151 aceid(s) already in the ACL completely\nreplace the ACE(s) in the ACL Resource.\n\nACEs provided in
4152 the update without aceid properties are added and\nassigned unique aceids in the ACL Resource.\n",
4153         "parameters": [
4154             { "$ref": "#/parameters/interface" },
4155             { "$ref": "#/parameters/ace-filtered" },
4156         ]
4157         "name": "body",
4158         "in": "body",
4159         "required": true,
4160         "schema": { "$ref": "#/definitions/Acl2-Update" },
4161         "x-example":
4162             {
4163                 "aclist2": [
4164                     {
4165                         "aceid": 1,
4166                         "subject": {
4167                             "authority": "484b8a51-cb23-46c0-a5f1-b4aebef50ebe",
4168                             "role": "SOME_STRING"
4169                         },
4170                         "resources": [
4171                             {
4172                                 "href": "/light"
4173                             },
4174                             {
4175                                 "href": "/door"
4176                             }
4177                         ],
4178                         "permission": 24
4179                     },
4180                     {
4181                         "aceid": 3,
4182                         "subject": {
4183                             "uuid": "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9"
4184                         },
4185                         "resources": [
4186                             {
4187                                 "href": "/light"
4188                             },
4189                             {
4190                                 "href": "/door"
4191                             }
4192                         ],
4193                         "permission": 24
4194                     }
4195                 ],
4196                 "rowneruuid": "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9"
4197             }
4198     }
4199 ],
4200     "responses": {
4201         "400": {
4202             "description": "The request is invalid."

```

```

4203     },
4204     "201": {
4205         "description": "The ACL entry is created."
4206     },
4207     "204": {
4208         "description": "The ACL entry is updated."
4209     }
4210 },
4211 },
4212 "delete": {
4213     "description": "Deletes ACL entries.\nWhen DELETE is used without query parameters, all the
4214 ACE entries are deleted.\nWhen DELETE is used with a query parameter, only the ACEs matching
4215 the\nspecified parameter are deleted.\n",
4216     "parameters": [
4217         {"$ref": "#/parameters/interface"},
4218         {"$ref": "#/parameters/ace-filtered"}
4219     ],
4220     "responses": {
4221         "200": {
4222             "description": "The matching ACEs or the entire ACL Resource has been successfully
4223 deleted."
4224         },
4225         "400": {
4226             "description": "The request is invalid."
4227         }
4228     }
4229 }
4230 },
4231 },
4232 "parameters": {
4233     "interface": {
4234         "in": "query",
4235         "name": "if",
4236         "type": "string",
4237         "enum": ["oic.if.baseline"]
4238     },
4239     "ace-filtered": {
4240         "in": "query",
4241         "name": "aceid",
4242         "required": false,
4243         "type": "integer",
4244         "description": "Only applies to the ACE with the specified aceid.",
4245         "x-example": 2112
4246     }
4247 },
4248 "definitions": {
4249     "Acl2": {
4250         "properties": {
4251             "owneruuid": {
4252                 "description": "The value identifies the unique Resource owner\nFormat pattern according
4253 to IETF RFC 4122.",
4254                 "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-
4255 9]{12}$",
4256                 "type": "string"
4257             },
4258             "rt": {
4259                 "description": "Resource Type of the Resource.",
4260                 "items": {
4261                     "maxLength": 64,
4262                     "type": "string",
4263                     "enum": ["oic.r.acl2"]
4264                 },
4265                 "minItems": 1,
4266                 "maxItems": 1,
4267                 "readOnly": true,
4268                 "type": "array"
4269             },
4270             "aclist2": {
4271                 "description": "Access Control Entries in the ACL Resource.",
4272                 "items": {
4273                     "properties": {

```



```

4274         "aceid": {
4275             "description": "An identifier for the ACE that is unique within the ACL. In cases
4276 where it isn't supplied in an update, the Server will add the ACE and assign it a unique value.",
4277             "minimum": 1,
4278             "type": "integer"
4279         },
4280         "permission": {
4281             "description": "Bitmask encoding of CRUDN permission\nThe encoded bitmask indicating
4282 permissions.",
4283             "x-detail-desc": [
4284                 "0 - No permissions",
4285                 "1 - Create permission is granted",
4286                 "2 - Read, observe, discover permission is granted",
4287                 "4 - Write, update permission is granted",
4288                 "8 - Delete permission is granted",
4289                 "16 - Notify permission is granted"
4290             ],
4291             "maximum": 31,
4292             "minimum": 0,
4293             "type": "integer"
4294         },
4295         "resources": {
4296             "description": "References the application's Resources to which a security policy
4297 applies.",
4298             "items": {
4299                 "description": "Each Resource must have at least one of these properties set.",
4300                 "properties": {
4301                     "href": {
4302                         "description": "When present, the ACE only applies when the href matches\nThis
4303 is the target URI, it can be specified as a Relative Reference or fully-qualified URI.",
4304                         "format": "uri",
4305                         "maxLength": 256,
4306                         "type": "string"
4307                     },
4308                     "wc": {
4309                         "description": "A wildcard matching policy.",
4310                         "pattern": "^[~+*]$",
4311                         "type": "string"
4312                     }
4313                 },
4314                 "type": "object"
4315             },
4316             "type": "array"
4317         },
4318         "subject": {
4319             "anyOf": [
4320                 {
4321                     "description": "This is the Device identifier.",
4322                     "properties": {
4323                         "uuid": {
4324                             "description": "A UUID Device ID\nFormat pattern according to IETF RFC
4325 4122.",
4326                             "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-
4327 fA-F0-9]{12}$",
4328                             "type": "string"
4329                         }
4330                     },
4331                     "required": [
4332                         "uuid"
4333                     ],
4334                     "type": "object"
4335                 },
4336                 {
4337                     "description": "Security role specified as an <Authority> & <Rolename>. A NULL
4338 <Authority> refers to the local entity or Device.",
4339                     "properties": {
4340                         "authority": {
4341                             "description": "The Authority component of the entity being identified. A
4342 NULL <Authority> refers to the local entity or Device.",
4343                             "type": "string"
4344                         }

```

```

4345         "role": {
4346             "description": "The ID of the role being identified.",
4347             "type": "string"
4348         }
4349     },
4350     "required": [
4351         "role"
4352     ],
4353     "type": "object"
4354 },
4355 {
4356     "properties": {
4357         "conntype": {
4358             "description": "This property allows an ACE to be matched based on the
4359 connection or message type.",
4360             "x-detail-desc": [
4361                 "auth-crypt - ACE applies if the Client is authenticated and the data
4362 channel or message is encrypted and integrity protected",
4363                 "anon-clear - ACE applies if the Client is not authenticated and the data
4364 channel or message is not encrypted but may be integrity protected"
4365             ],
4366             "enum": [
4367                 "auth-crypt",
4368                 "anon-clear"
4369             ],
4370             "type": "string"
4371         }
4372     },
4373     "required": [
4374         "conntype"
4375     ],
4376     "type": "object"
4377 }
4378 ],
4379 },
4380 "validity": {
4381     "description": "validity is an array of time-pattern objects.",
4382     "items": {
4383         "description": "The time-pattern contains a period and recurrence expressed in
4384 RFC5545 syntax.",
4385         "properties": {
4386             "period": {
4387                 "description": "String represents a period using the RFC5545 Period.",
4388                 "type": "string"
4389             },
4390             "recurrence": {
4391                 "description": "String array represents a recurrence rule using the RFC5545
4392 Recurrence.",
4393                 "items": {
4394                     "type": "string"
4395                 },
4396                 "type": "array"
4397             }
4398         },
4399         "required": [
4400             "period"
4401         ],
4402         "type": "object"
4403     },
4404     "type": "array"
4405 }
4406 },
4407 "required": [
4408     "aceid",
4409     "resources",
4410     "permission",
4411     "subject"
4412 ],
4413 "type": "object"
4414 },
4415 "type": "array"

```

```

4416     },
4417     "n": {
4418       "$ref":
4419       "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
4420       schema.json#/definitions/n"
4421     },
4422     "id": {
4423       "$ref":
4424       "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
4425       schema.json#/definitions/id"
4426     },
4427     "if" : {
4428       "description": "The interface set supported by this Resource.",
4429       "items": {
4430         "enum": [
4431           "oic.if.baseline"
4432         ],
4433         "type": "string"
4434       },
4435       "minItems": 1,
4436       "maxItems": 1,
4437       "readOnly": true,
4438       "type": "array"
4439     }
4440   },
4441   "type" : "object",
4442   "required": ["aclist2", "rowneruuid"]
4443 },
4444 "Acl2-Update" : {
4445   "properties": {
4446     "rowneruuid" : {
4447       "description": "The value identifies the unique Resource owner\n Format pattern according
4448 to IETF RFC 4122.",
4449       "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-
4450 9]{12}$",
4451       "type": "string"
4452     },
4453     "aclist2" : {
4454       "description": "Access Control Entries in the ACL Resource.",
4455       "items": {
4456         "properties": {
4457           "aceid": {
4458             "description": "An identifier for the ACE that is unique within the ACL. In cases
4459 where it isn't supplied in an update, the Server will add the ACE and assign it a unique value.",
4460             "minimum": 1,
4461             "type": "integer"
4462           },
4463           "permission": {
4464             "description": "Bitmask encoding of CRUDN permission\nThe encoded bitmask indicating
4465 permissions.",
4466             "x-detail-desc": [
4467               "0 - No permissions",
4468               "1 - Create permission is granted",
4469               "2 - Read, observe, discover permission is granted",
4470               "4 - Write, update permission is granted",
4471               "8 - Delete permission is granted",
4472               "16 - Notify permission is granted"
4473             ],
4474             "maximum": 31,
4475             "minimum": 0,
4476             "type": "integer"
4477           },
4478           "resources": {
4479             "description": "References the application's Resources to which a security policy
4480 applies.",
4481             "items": {
4482               "description": "Each Resource must have at least one of these properties set.",
4483               "properties": {
4484                 "href": {
4485                   "description": "When present, the ACE only applies when the href matches\nThis
4486 is the target URI, it can be specified as a Relative Reference or fully-qualified URI.",

```

```

4487         "format": "uri",
4488         "maxLength": 256,
4489         "type": "string"
4490     },
4491     "wc": {
4492         "description": "A wildcard matching policy.",
4493         "x-detail-desc": [
4494             "+ - Matches all discoverable Resources",
4495             "- - Matches all non-discoverable Resources",
4496             "* - Matches all Resources"
4497         ],
4498         "enum": [
4499             "+",
4500             "-",
4501             "*"
4502         ],
4503         "type": "string"
4504     }
4505 },
4506 "type": "object"
4507 },
4508 "type": "array"
4509 },
4510 "subject": {
4511     "anyOf": [
4512         {
4513             "description": "This is the Device identifier.",
4514             "properties": {
4515                 "uuid": {
4516                     "description": "A UUID Device ID\n Format pattern according to IETF RFC
4122.",
4517                     "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-
4518 fA-F0-9]{12}$",
4519                     "type": "string"
4520                 }
4521             },
4522             "required": [
4523                 "uuid"
4524             ],
4525             "type": "object"
4526         },
4527         {
4528             "description": "Security role specified as an <Authority> & <Rolename>. A NULL
4529 <Authority> refers to the local entity or Device.",
4530             "properties": {
4531                 "authority": {
4532                     "description": "The Authority component of the entity being identified. A
4533 NULL <Authority> refers to the local entity or Device.",
4534                     "type": "string"
4535                 },
4536                 "role": {
4537                     "description": "The ID of the role being identified.",
4538                     "type": "string"
4539                 }
4540             },
4541             "required": [
4542                 "role"
4543             ],
4544             "type": "object"
4545         },
4546         {
4547             "properties": {
4548                 "conntype": {
4549                     "description": "This property allows an ACE to be matched based on the
4550 connection or message type.",
4551                     "x-detail-desc": [
4552                         "auth-crypt - ACE applies if the Client is authenticated and the data
4553 channel or message is encrypted and integrity protected",
4554                         "anon-clear - ACE applies if the Client is not authenticated and the data
4555 channel or message is not encrypted but may be integrity protected"
4556                     ],
4557                     "type": "string"
4558                 }
4559             }
4560         }
4561     ]
4562 }

```

```

4558         "enum": [
4559             "auth-crypt",
4560             "anon-clear"
4561         ],
4562         "type": "string"
4563     }
4564 },
4565     "required": [
4566         "conntype"
4567     ],
4568     "type": "object"
4569 }
4570 ]
4571 },
4572 "validity": {
4573     "description": "validity is an array of time-pattern objects.",
4574     "items": {
4575         "description": "The time-pattern contains a period and recurrence expressed in
RFC5545 syntax.",
4576         "properties": {
4577             "period": {
4578                 "description": "String represents a period using the RFC5545 Period.",
4579                 "type": "string"
4580             },
4581             "recurrence": {
4582                 "description": "String array represents a recurrence rule using the RFC5545
Recurrence.",
4583                 "items": {
4584                     "type": "string"
4585                 },
4586                 "type": "array"
4587             }
4588         }
4589     },
4590     "required": [
4591         "period"
4592     ],
4593     "type": "object"
4594 },
4595     "type": "array"
4596 }
4597 },
4598 "required": [
4599     "resources",
4600     "permission",
4601     "subject"
4602 ],
4603     "type": "object"
4604 },
4605     "type": "array"
4606 }
4607 }
4608 },
4609     "type": "object"
4610 }
4611 }
4612 }
4613

```

4614 C.2.5 Property definition

4615 Table C-1 defines the Properties that are part of the "oic.r.acl2" Resource Type.

4616 **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.

4617 **C.2.6 CRUDN behaviour**

4618 Table C-2 defines the CRUDN operations that are supported on the "oic.r.acl2" Resource Type.

4619 **Table C-2 – The CRUDN operations of the Resource with type "rt" = "oic.r.acl2".**

Create	Read	Update	Delete	Notify
	get	post	delete	observe

4620 **C.3 Credential**

4621 **C.3.1 Introduction**

4622 This Resource specifies credentials a Device may use to establish secure communication.

4623 Retrieves the credential data.

4624 When used without query parameters, all the credential entries are returned.

4625 When used with a query parameter, only the credentials matching the specified
4626 parameter are returned.

4627
4628 Note that write-only credential data will not be returned.
4629

4630 **C.3.2 Well-known URI**

4631 /oic/sec/cred

4632 **C.3.3 Resource type**

4633 The Resource Type is defined as: "oic.r.cred".

4634 **C.3.4 OpenAPI 2.0 definition**

```
4635 {
4636   "swagger": "2.0",
4637   "info": {
```

```

4638     "title": "Credential",
4639     "version": "v1.0-20181031",
4640     "license": {
4641         "name": "OCF Data Model License",
4642         "url":
4643     "https://github.com/openconnectivityfoundation/core/blob/e28a9e0a92e17042ba3e83661e4c0fbce8bdc4ba/LI
4644     CENSE.md",
4645         "x-copyright": "copyright 2016-2017, 2019 Open Connectivity Foundation, Inc. All rights
4646     reserved."
4647     },
4648     "termsOfService": "https://openconnectivityfoundation.github.io/core/DISCLAIMER.md"
4649     },
4650     "schemes": ["http"],
4651     "consumes": ["application/json"],
4652     "produces": ["application/json"],
4653     "paths": {
4654         "/oic/sec/cred" : {
4655             "get": {
4656                 "description": "This Resource specifies credentials a Device may use to establish secure
4657     communication.\nRetrieves the credential data.\nWhen used without query parameters, all the
4658     credential entries are returned.\nWhen used with a query parameter, only the credentials matching
4659     the specified\nparameter are returned.\n\nNote that write-only credential data will not be
4660     returned.\n",
4661                 "parameters": [
4662                     {"$ref": "#/parameters/interface"}
4663                     , {"$ref": "#/parameters/cred-filtered-credid"}
4664                     , {"$ref": "#/parameters/cred-filtered-subjectuuid"}
4665                 ],
4666                 "responses": {
4667                     "200": {
4668                         "description": "",
4669                         "x-example":
4670                             {
4671                                 "xt": ["oic.r.cred"],
4672                                 "creds": [
4673                                     {
4674                                         "credid": 55,
4675                                         "subjectuuid": "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
4676                                         "roleid": {
4677                                             "authority": "484b8a51-cb23-46c0-a5f1-b4aebef50ebe",
4678                                             "role": "SOME_STRING"
4679                                         },
4680                                         "credtype": 32,
4681                                         "publicdata": {
4682                                             "encoding": "oic.sec.encoding.pem",
4683                                             "data": "PEM-ENCODED-VALUE"
4684                                         },
4685                                         "privatedata": {
4686                                             "encoding": "oic.sec.encoding.raw",
4687                                             "data": "RAW-ENCODED-VALUE",
4688                                             "handle": 4
4689                                         },
4690                                         "optionaldata": {
4691                                             "revstat": false,
4692                                             "encoding": "oic.sec.encoding.pem",
4693                                             "data": "PEM-ENCODED-VALUE"
4694                                         },
4695                                         "period": "20160101T180000Z/20170102T070000Z",
4696                                         "crms": [ "oic.sec.crm.pk10" ]
4697                                     },
4698                                     {
4699                                         "credid": 56,
4700                                         "subjectuuid": "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
4701                                         "roleid": {
4702                                             "authority": "484b8a51-cb23-46c0-a5f1-b4aebef50ebe",
4703                                             "role": "SOME_STRING"
4704                                         },
4705                                         "credtype": 1,
4706                                         "publicdata": {
4707                                             "encoding": "oic.sec.encoding.pem",
4708                                             "data": "PEM-ENCODED-VALUE"

```

```

4709         },
4710         "privatedata": {
4711             "encoding": "oic.sec.encoding.base64",
4712             "data": "BASE-64-ENCODED-VALUE",
4713             "handle": 4
4714         },
4715         "optionaldata": {
4716             "revstat": false,
4717             "encoding": "oic.sec.encoding.pem",
4718             "data": "PEM-ENCODED-VALUE"
4719         },
4720         "period": "20160101T180000Z/20170102T070000Z",
4721         "crms": [ "oic.sec.crm.pk10" ]
4722     }
4723 ],
4724     "xowneruuid": "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9"
4725 }
4726 ,
4727     "schema": { "$ref": "#/definitions/Cred" }
4728 },
4729     "400": {
4730         "description": "The request is invalid."
4731     }
4732 }
4733 },
4734     "post": {
4735         "description": "Updates the credential Resource with the provided
4736 credentials.\n\nCredentials provided in the update with credid(s) not currently in the\ncredential
4737 Resource are added.\n\nCredentials provided in the update with credid(s) already in the\ncredential
4738 Resource completely replace the creds in the credential\nResource.\n\nCredentials provided in the
4739 update without credid(s) properties are\nadded and assigned unique credid(s) in the credential
4740 Resource.\n",
4741         "parameters": [
4742             { "$ref": "#/parameters/interface" },
4743             {
4744                 "name": "body",
4745                 "in": "body",
4746                 "required": true,
4747                 "schema": { "$ref": "#/definitions/Cred-Update" },
4748                 "x-example":
4749                 {
4750                     "creds": [
4751                         {
4752                             "credid": 55,
4753                             "subjectuuid": "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
4754                             "roleid": {
4755                                 "authority": "484b8a51-cb23-46c0-a5f1-b4aebef50ebe",
4756                                 "role": "SOME_STRING"
4757                             },
4758                             "credtype": 32,
4759                             "publicdata": {
4760                                 "encoding": "oic.sec.encoding.pem",
4761                                 "data": "PEM-ENCODED-VALUE"
4762                             },
4763                             "privatedata": {
4764                                 "encoding": "oic.sec.encoding.raw",
4765                                 "data": "RAW-ENCODED-VALUE",
4766                                 "handle": 4
4767                             },
4768                             "optionaldata": {
4769                                 "revstat": false,
4770                                 "encoding": "oic.sec.encoding.pem",
4771                                 "data": "PEM-ENCODED-VALUE"
4772                             },
4773                             "period": "20160101T180000Z/20170102T070000Z",
4774                             "crms": [ "oic.sec.crm.pk10" ]
4775                         },
4776                         {
4777                             "credid": 56,
4778                             "subjectuuid": "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
4779                             "roleid": {

```



```

4780         "authority": "484b8a51-cb23-46c0-a5f1-b4aebef50ebe",
4781         "role": "SOME_STRING"
4782     },
4783     "credtype": 1,
4784     "publicdata": {
4785         "encoding": "oic.sec.encoding.pem",
4786         "data": "PEM-ENCODED-VALUE"
4787     },
4788     "privatedata": {
4789         "encoding": "oic.sec.encoding.base64",
4790         "data": "BASE-64-ENCODED-VALUE",
4791         "handle": 4
4792     },
4793     "optionaldata": {
4794         "revstat": false,
4795         "encoding": "oic.sec.encoding.pem",
4796         "data": "PEM-ENCODED-VALUE"
4797     },
4798     "period": "20160101T180000Z/20170102T070000Z",
4799     "crms": [ "oic.sec.crm.pk10" ]
4800 }
4801 ],
4802     "rowneruuid": "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9"
4803 }
4804 }
4805 ],
4806 "responses": {
4807     "400": {
4808         "description": "The request is invalid."
4809     },
4810     "201": {
4811         "description": "The credential entry is created."
4812     },
4813     "204": {
4814         "description": "The credential entry is updated."
4815     }
4816 }
4817 },
4818 "delete": {
4819     "description": "Deletes credential entries.\nWhen DELETE is used without query parameters,
4820 all the cred entries are deleted.\nWhen DELETE is used with a query parameter, only the entries
4821 matching\nthe query parameter are deleted.\n",
4822     "parameters": [
4823         {"$ref": "#/parameters/interface"},
4824         {"$ref": "#/parameters/cred-filtered-credid"},
4825         {"$ref": "#/parameters/cred-filtered-subjectuuid"}
4826     ],
4827     "responses": {
4828         "400": {
4829             "description": "The request is invalid."
4830         },
4831         "204": {
4832             "description": "The specific credential(s) or the the entire credential Resource has
4833 been successfully deleted."
4834         }
4835     }
4836 }
4837 }
4838 },
4839 "parameters": {
4840     "interface": {
4841         "in": "query",
4842         "name": "if",
4843         "type": "string",
4844         "enum": ["oic.if.baseline"]
4845     },
4846     "cred-filtered-credid": {
4847         "in": "query",
4848         "name": "credid",
4849         "required": false,
4850         "type": "integer",

```

```

4851         "description" : "Only applies to the credential with the specified credid.",
4852         "x-example" : 2112
4853     },
4854     "cred-filtered-subjectuuid" : {
4855         "in" : "query",
4856         "name" : "subjectuuid",
4857         "required" : false,
4858         "type" : "string",
4859         "description" : "Only applies to credentials with the specified subject UUID.",
4860         "x-example" : "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9"
4861     }
4862 },
4863 "definitions": {
4864     "Cred" : {
4865         "properties": {
4866             "rowneruuid" : {
4867                 "description": "Format pattern according to IETF RFC 4122.",
4868                 "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}$",
4869                 "type": "string"
4870             },
4871         },
4872         "rt" : {
4873             "description": "Resource Type of the Resource.",
4874             "items": {
4875                 "maxLength": 64,
4876                 "type": "string",
4877                 "enum": ["oic.r.cred"]
4878             },
4879             "minItems": 1,
4880             "readOnly": true,
4881             "type": "array",
4882             "uniqueItems": true
4883         },
4884         "n": {
4885             "$ref":
4886 "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
4887 schema.json#/definitions/n"
4888         },
4889         "id": {
4890             "$ref":
4891 "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
4892 schema.json#/definitions/id"
4893         },
4894         "creds" : {
4895             "description": "List of credentials available at this Resource.",
4896             "items": {
4897                 "properties": {
4898                     "credid": {
4899                         "description": "Local reference to a credential Resource.",
4900                         "type": "integer"
4901                     },
4902                     "credtype": {
4903                         "description": "Representation of this credential's type\nCredential Types - Cred
4904 type encoded as a bitmask.0 - Empty credential used for testing1 - Symmetric pair-wise key2 -
4905 Symmetric group key4 - Asymmetric signing key8 - Asymmetric signing key with certificatel6 - PIN or
4906 password32 - Asymmetric encryption key.",
4907                         "maximum": 63,
4908                         "minimum": 0,
4909                         "type": "integer"
4910                     },
4911                     "credusage": {
4912                         "description": "A string that provides hints about how/where the cred is used\nThe
4913 type of credusage.oic.sec.cred.trustca - Trust certificateoic.sec.cred.cert -
4914 Certificateoic.sec.cred.rolecert - Role Certificateoic.sec.cred.mfgtrustca - Manufacturer
4915 Certificate Trust Anchoroic.sec.cred.mfgcert - Manufacturer Certificate.",
4916                         "enum": [
4917                             "oic.sec.cred.trustca",
4918                             "oic.sec.cred.cert",
4919                             "oic.sec.cred.rolecert",
4920                             "oic.sec.cred.mfgtrustca",
4921                             "oic.sec.cred.mfgcert"

```

```

4922         ],
4923         "type": "string"
4924     },
4925     "crms": {
4926         "description": "The refresh methods that may be used to update this credential.",
4927         "items": {
4928             "description": "Each enum represents a method by which the credentials are
4929 refreshed.oic.sec.crm.pro - Credentials refreshed by a provisioning serviceoic.sec.crm.rdp -
4930 Credentials refreshed by a key agreement protocol and random PINoic.sec.crm.psk - Credentials
4931 refreshed by a key agreement protocoloic.sec.crm.skdc - Credentials refreshed by a key distribution
4932 serviceoic.sec.crm.pk10 - Credentials refreshed by a PKCS#10 request to a CA.",
4933             "enum": [
4934                 "oic.sec.crm.pro",
4935                 "oic.sec.crm.psk",
4936                 "oic.sec.crm.rdp",
4937                 "oic.sec.crm.skdc",
4938                 "oic.sec.crm.pk10"
4939             ],
4940             "type": "string"
4941         },
4942         "type": "array",
4943         "uniqueItems": true
4944     },
4945     "optionaldata": {
4946         "description": "Credential revocation status information\nOptional credential
4947 contents describes revocation status for this credential.",
4948         "properties": {
4949             "data": {
4950                 "description": "The encoded structure.",
4951                 "type": "string"
4952             },
4953             "encoding": {
4954                 "description": "A string specifying the encoding format of the data contained in
4955 the optdata.",
4956                 "x-detail-desc": [
4957                     "oic.sec.encoding.pem - Encoding for PEM encoded certificate or chain."
4958                 ],
4959                 "enum": [
4960                     "oic.sec.encoding.pem"
4961                 ],
4962                 "type": "string"
4963             },
4964             "revstat": {
4965                 "description": "Revocation status flag - true = revoked.",
4966                 "type": "boolean"
4967             }
4968         },
4969         "required": [
4970             "revstat"
4971         ],
4972         "type": "object"
4973     },
4974     "period": {
4975         "description": "String with RFC5545 Period.",
4976         "type": "string"
4977     },
4978     "privatedata": {
4979         "description": "Private credential information\nCredential Resource non-public
4980 contents.",
4981         "properties": {
4982             "data": {
4983                 "description": "The encoded value.",
4984                 "maxLength": 3072,
4985                 "type": "string"
4986             },
4987             "encoding": {
4988                 "description": "A string specifying the encoding format of the data contained in
4989 the privdata.",
4990                 "x-detail-desc": [
4991                     "oic.sec.encoding.pem - Encoding for PEM encoded private key.",
4992                     "oic.sec.encoding.base64 - Encoding for Base64 encoded PSK.",

```

```

4993         "oic.sec.encoding.handle - Data is contained in a storage sub-system
4994 referenced using a handle.",
4995         "oic.sec.encoding.raw - Raw hex encoded data."
4996     ],
4997     "enum": [
4998         "oic.sec.encoding.pem",
4999         "oic.sec.encoding.base64",
5000         "oic.sec.encoding.handle",
5001         "oic.sec.encoding.raw"
5002     ],
5003     "type": "string"
5004 },
5005 "handle": {
5006     "description": "Handle to a key storage Resource.",
5007     "type": "integer"
5008 }
5009 },
5010 "required": [
5011     "encoding"
5012 ],
5013 "type": "object"
5014 },
5015 "publicdata": {
5016     "description": "Public credential information.",
5017     "properties": {
5018         "data": {
5019             "description": "The encoded value.",
5020             "maxLength": 3072,
5021             "type": "string"
5022         },
5023         "encoding": {
5024             "description": "A string specifying the encoding format of the data contained in
5025 the pubdata.",
5026             "x-detail-desc": [
5027                 "oic.sec.encoding.pem - Encoding for PEM encoded certificate or chain."
5028             ],
5029             "enum": [
5030                 "oic.sec.encoding.pem"
5031             ],
5032             "type": "string"
5033         }
5034     },
5035     "type": "object"
5036 },
5037 "roleid": {
5038     "description": "The role this credential possesses\nSecurity role specified as an
5039 <Authority> & <Rolename>. A NULL <Authority> refers to the local entity or Device.",
5040     "properties": {
5041         "authority": {
5042             "description": "The Authority component of the entity being identified. A NULL
5043 <Authority> refers to the local entity or Device.",
5044             "type": "string"
5045         },
5046         "role": {
5047             "description": "The ID of the role being identified.",
5048             "type": "string"
5049         }
5050     },
5051     "required": [
5052         "role"
5053     ],
5054     "type": "object"
5055 },
5056 "subjectuuid": {
5057     "anyOf": [
5058         {
5059             "description": "The id of the Device, which the cred entry applies to or \"*\n
5060 for wildcard identity.",
5061             "pattern": "^\\*?$",
5062             "type": "string"
5063         }
5064     ],

```

```

5064         {
5065             "description": "Format pattern according to IETF RFC 4122.",
5066             "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-
5067 F0-9]{12}$",
5068             "type": "string"
5069         }
5070     ]
5071 }
5072 },
5073 "type": "object"
5074 },
5075 "type": "array"
5076 },
5077 "if" : {
5078     "description": "The interface set supported by this Resource.",
5079     "items": {
5080         "enum": [
5081             "oic.if.baseline"
5082         ],
5083         "type": "string"
5084     },
5085     "minItems": 1,
5086     "readOnly": true,
5087     "type": "array"
5088 }
5089 },
5090 "type" : "object",
5091 "required": ["creds", "rowneruuid"]
5092 },
5093 "Cred-Update" : {
5094     "properties": {
5095         "rowneruuid" : {
5096             "description": "Format pattern according to IETF RFC 4122.",
5097             "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-
5098 9]{12}$",
5099             "type": "string"
5100         },
5101         "creds" : {
5102             "description": "List of credentials available at this Resource.",
5103             "items": {
5104                 "properties": {
5105                     "credid": {
5106                         "description": "Local reference to a credential Resource.",
5107                         "type": "integer"
5108                     },
5109                     "credtype": {
5110                         "description": "Representation of this credential's type\nCredential Types - Cred
5111 type encoded as a bitmask.0 - Empty credential used for testing1 - Symmetric pair-wise key2 -
5112 Symmetric group key4 - Asymmetric signing key8 - Asymmetric signing key with certificate16 - PIN or
5113 password32 - Asymmetric encryption key.",
5114                         "maximum": 63,
5115                         "minimum": 0,
5116                         "type": "integer"
5117                     },
5118                     "credusage": {
5119                         "description": "A string that provides hints about how/where the cred is used\nThe
5120 type of credusage.oic.sec.cred.trustca - Trust certificateoic.sec.cred.cert -
5121 Certificateoic.sec.cred.rolecert - Role Certificateoic.sec.cred.mfgtrustca - Manufacturer
5122 Certificate Trust Anchoroic.sec.cred.mfgcert - Manufacturer Certificate.",
5123                         "enum": [
5124                             "oic.sec.cred.trustca",
5125                             "oic.sec.cred.cert",
5126                             "oic.sec.cred.rolecert",
5127                             "oic.sec.cred.mfgtrustca",
5128                             "oic.sec.cred.mfgcert"
5129                         ],
5130                         "type": "string"
5131                     },
5132                     "crms": {
5133                         "description": "The refresh methods that may be used to update this credential.",
5134                         "items": {

```

```

5135         "description": "Each enum represents a method by which the credentials are
5136 refreshed.oic.sec.crm.pro - Credentials refreshed by a provisioning serviceoic.sec.crm.rdp -
5137 Credentials refreshed by a key agreement protocol and random PINoic.sec.crm.psk - Credentials
5138 refreshed by a key agreement protocoloic.sec.crm.skdc - Credentials refreshed by a key distribution
5139 serviceoic.sec.crm.pk10 - Credentials refreshed by a PKCS#10 request to a CA.",
5140         "enum": [
5141             "oic.sec.crm.pro",
5142             "oic.sec.crm.psk",
5143             "oic.sec.crm.rdp",
5144             "oic.sec.crm.skdc",
5145             "oic.sec.crm.pk10"
5146         ],
5147         "type": "string"
5148     },
5149     "type": "array"
5150 },
5151 "optionaldata": {
5152     "description": "Credential revocation status information\nOptional credential
5153 contents describes revocation status for this credential.",
5154     "properties": {
5155         "data": {
5156             "description": "The encoded structure.",
5157             "type": "string"
5158         },
5159         "encoding": {
5160             "description": "A string specifying the encoding format of the data contained in
5161 the optdata.",
5162             "x-detail-desc": [
5163                 "oic.sec.encoding.pem - Encoding for PEM encoded certificate or chain."
5164             ],
5165             "enum": [
5166                 "oic.sec.encoding.pem"
5167             ],
5168             "type": "string"
5169         },
5170         "revstat": {
5171             "description": "Revocation status flag - true = revoked.",
5172             "type": "boolean"
5173         }
5174     },
5175     "required": [
5176         "revstat"
5177     ],
5178     "type": "object"
5179 },
5180 "period": {
5181     "description": "String with RFC5545 Period.",
5182     "type": "string"
5183 },
5184 "privatedata": {
5185     "description": "Private credential information\nCredential Resource non-public
5186 contents.",
5187     "properties": {
5188         "data": {
5189             "description": "The encoded value.",
5190             "maxLength": 3072,
5191             "type": "string"
5192         },
5193         "encoding": {
5194             "description": "A string specifying the encoding format of the data contained in
5195 the privdata.",
5196             "x-detail-desc": [
5197                 "oic.sec.encoding.pem - Encoding for PEM encoded private key.",
5198                 "oic.sec.encoding.base64 - Encoding for Base64 encoded PSK.",
5199                 "oic.sec.encoding.handle - Data is contained in a storage sub-system
5200 referenced using a handle.",
5201                 "oic.sec.encoding.raw - Raw hex encoded data."
5202             ],
5203             "enum": [
5204                 "oic.sec.encoding.pem",
5205                 "oic.sec.encoding.base64",

```

```

5206         "oic.sec.encoding.handle",
5207         "oic.sec.encoding.raw"
5208     ],
5209     "type": "string"
5210 },
5211 "handle": {
5212     "description": "Handle to a key storage Resource.",
5213     "type": "integer"
5214 }
5215 },
5216 "required": [
5217     "encoding"
5218 ],
5219 "type": "object"
5220 },
5221 "publicdata": {
5222     "properties": {
5223         "data": {
5224             "description": "The encoded value.",
5225             "maxLength": 3072,
5226             "type": "string"
5227         },
5228         "encoding": {
5229             "description": "Public credential information\nA string specifying the encoding
5230 format of the data contained in the pubdata.",
5231             "x-detail-desc": [
5232                 "oic.sec.encoding.pem - Encoding for PEM encoded certificate or chain."
5233             ],
5234             "enum": [
5235                 "oic.sec.encoding.pem"
5236             ],
5237             "type": "string"
5238         }
5239     },
5240     "type": "object"
5241 },
5242 "roleid": {
5243     "description": "The role this credential possesses\nSecurity role specified as an
5244 <Authority> & <Rolename>. A NULL <Authority> refers to the local entity or Device.",
5245     "properties": {
5246         "authority": {
5247             "description": "The Authority component of the entity being identified. A NULL
5248 <Authority> refers to the local entity or Device.",
5249             "type": "string"
5250         },
5251         "role": {
5252             "description": "The ID of the role being identified.",
5253             "type": "string"
5254         }
5255     },
5256     "required": [
5257         "role"
5258     ],
5259     "type": "object"
5260 },
5261 "subjectuuid": {
5262     "anyOf": [
5263         {
5264             "description": "The id of the Device, which the cred entry applies to or \"*\n
5265 for wildcard identity.",
5266             "pattern": "^\\*$",
5267             "type": "string"
5268         },
5269         {
5270             "description": "Format pattern according to IETF RFC 4122.",
5271             "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-
5272 F0-9]{12}$",
5273             "type": "string"
5274         }
5275     ]
5276 }

```

```

5277     },
5278     "type": "object"
5279   },
5280   "type": "array"
5281 },
5282 "if" :
5283 {
5284   "description": "The interface set supported by this Resource.",
5285   "items": {
5286     "enum": [
5287       "oic.if.baseline"
5288     ],
5289     "type": "string"
5290   },
5291   "minItems": 1,
5292   "readOnly": true,
5293   "type": "array"
5294 }
5295 },
5296 "type" : "object"
5297 }
5298 }
5299 }
5300

```

5301 C.3.5 Property definition

5302 Table C-3 defines the Properties that are part of the "oic.r.cred" Resource Type.

5303 **Table C-3 – The Property definitions of the Resource with type "rt" = "oic.r.cred".**

Property name	Value type	Mandatory	Access mode	Description
rowneruuid	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.
rowneruuid	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.

5304 C.3.6 CRUDN behaviour

5305 Table C-4 defines the CRUDN operations that are supported on the "oic.r.cred" Resource Type.

5306 **Table C-4 – The CRUDN operations of the Resource with type "rt" = "oic.r.cred".**

Create	Read	Update	Delete	Notify
	get	post	delete	observe

5307 **C.4 Certificate Signing Request**

5308 **C.4.1 Introduction**

5309 This Resource specifies a Certificate Signing Request.
5310

5311 **C.4.2 Well-known URI**

5312 /oic/sec/csr

5313 **C.4.3 Resource type**

5314 The Resource Type is defined as: "oic.r.csr".

5315 **C.4.4 OpenAPI 2.0 definition**

```

5316 {
5317   "swagger": "2.0",
5318   "info": {
5319     "title": "Certificate Signing Request",
5320     "version": "v1.0-20150819",
5321     "license": {
5322       "name": "OCF Data Model License",
5323       "url":
5324         "https://github.com/openconnectivityfoundation/core/blob/e28a9e0a92e17042ba3e83661e4c0fbce8bdc4ba/LI
5325         CENSE.md",
5326       "x-copyright": "copyright 2016-2017, 2019 Open Connectivity Foundation, Inc. All rights
5327         reserved."
5328     },
5329     "termsOfService": "https://openconnectivityfoundation.github.io/core/DISCLAIMER.md"
5330   },
5331   "schemes": ["http"],
5332   "consumes": ["application/json"],
5333   "produces": ["application/json"],
5334   "paths": {
5335     "/oic/sec/csr" : {
5336       "get": {
5337         "description": "This Resource specifies a Certificate Signing Request.\n",
5338         "parameters": [
5339           {"$ref": "#/parameters/interface"}
5340         ],
5341         "responses": {
5342           "200": {
5343             "description": "",
5344             "x-example":
5345               {
5346                 "rt": ["oic.r.csr"],
5347                 "encoding" : "oic.sec.encoding.pem",
5348                 "csr": "PEMENCODEDCSR"
5349               },
5350             "schema": { "$ref": "#/definitions/Csr" }
5351           },
5352           "404": {
5353             "description": "The Device does not support certificates and generating CSRs."
5354           },
5355           "503": {
5356             "description": "The Device is not yet ready to return a response. Try again later."
5357           }
5358         }
5359       }
5360     }
5361   },
5362   "parameters": {

```

```

5363     "interface" : {
5364         "in" : "query",
5365         "name" : "if",
5366         "type" : "string",
5367         "enum" : ["oic.if.baseline"]
5368     }
5369 },
5370 "definitions": {
5371     "Csr" : {
5372         "properties": {
5373             "rt" : {
5374                 "description": "Resource Type of the Resource.",
5375                 "items": {
5376                     "maxLength": 64,
5377                     "type": "string",
5378                     "enum": ["oic.r.csr"]
5379                 },
5380                 "minItems": 1,
5381                 "readOnly": true,
5382                 "type": "array"
5383             },
5384             "encoding": {
5385                 "description": "A string specifying the encoding format of the data contained in CSR.",
5386                 "x-detail-desc": [
5387                     "oic.sec.encoding.pem - Encoding for PEM encoded CSR."
5388                 ],
5389                 "enum": [
5390                     "oic.sec.encoding.pem"
5391                 ],
5392                 "readOnly": true,
5393                 "type": "string"
5394             },
5395             "n": {
5396                 "$ref":
5397 "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
5398 schema.json#/definitions/n"
5399             },
5400             "id": {
5401                 "$ref":
5402 "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
5403 schema.json#/definitions/id"
5404             },
5405             "csr": {
5406                 "description": "Signed CSR in ASN.1 in the encoding specified by the encoding property.",
5407                 "maxLength": 3072,
5408                 "readOnly": true,
5409                 "type": "string"
5410             },
5411             "if": {
5412                 "description": "The interface set supported by this Resource.",
5413                 "items": {
5414                     "enum": [
5415                         "oic.if.baseline"
5416                     ],
5417                     "type": "string"
5418                 },
5419                 "minItems": 1,
5420                 "readOnly": true,
5421                 "type": "array"
5422             }
5423         },
5424         "type" : "object",
5425         "required": ["csr", "encoding"]
5426     }
5427 }
5428 }
5429

```

5430 C.4.5 Property definition

5431 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.

5433 C.4.6 CRUDN behaviour

5434 Table C-6 defines the CRUDN operations that are supported on the "oic.r.csr" Resource Type.

5435 Table C-6 – The CRUDN operations of the Resource with type "rt" = "oic.r.csr".

Create	Read	Update	Delete	Notify
	get			observe

5436 C.5 Device Owner Transfer Method**5437 C.5.1 Introduction**

5438 This Resource specifies properties needed to establish a Device owner.

5439

5440 C.5.2 Well-known URI

5441 /oic/sec/doxm

5442 C.5.3 Resource type

5443 The Resource Type is defined as: "oic.r.doxm".

5444 C.5.4 OpenAPI 2.0 definition

```

5445 {
5446   "swagger": "2.0",
5447   "info": {
5448     "title": "Device Owner Transfer Method",
5449     "version": "v1.0-20181001",
5450     "license": {
5451       "name": "OCF Data Model License",
5452       "url":
5453         "https://github.com/openconnectivityfoundation/core/blob/e28a9e0a92e17042ba3e83661e4c0fbce8bdc4ba/LI
5454         CENSE.md",
5455       "x-copyright": "copyright 2016-2017, 2019 Open Connectivity Foundation, Inc. All rights

```

```

5456 reserved."
5457 },
5458 "termsOfService": "https://openconnectivityfoundation.github.io/core/DISCLAIMER.md"
5459 },
5460 "schemes": ["http"],
5461 "consumes": ["application/json"],
5462 "produces": ["application/json"],
5463 "paths": {
5464   "/oic/sec/doxm" : {
5465     "get": {
5466       "description": "This Resource specifies properties needed to establish a Device owner.\n",
5467       "parameters": [
5468         { "$ref": "#/parameters/interface" }
5469       ],
5470       "responses": {
5471         "200": {
5472           "description": "",
5473           "x-example":
5474             {
5475               "rt": ["oic.r.doxm"],
5476               "oxms": [ 0, 2, 3 ],
5477               "oxmsel": 0,
5478               "sct": 16,
5479               "owned": true,
5480               "deviceuuid": "de305d54-75b4-431b-adb2-eb6b9e546014",
5481               "devowneruuid": "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
5482               "rowneruuid": "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9"
5483             }
5484           ,
5485           "schema": { "$ref": "#/definitions/Doxm" }
5486         },
5487         "400": {
5488           "description": "The request is invalid."
5489         }
5490       }
5491     },
5492     "post": {
5493       "description": "Updates the DOXM Resource data.\n",
5494       "parameters": [
5495         { "$ref": "#/parameters/interface" },
5496         {
5497           "name": "body",
5498           "in": "body",
5499           "required": true,
5500           "schema": { "$ref": "#/definitions/Doxm-Update" },
5501           "x-example":
5502             {
5503               "oxmsel": 0,
5504               "owned": true,
5505               "deviceuuid": "de305d54-75b4-431b-adb2-eb6b9e546014",
5506               "devowneruuid": "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9",
5507               "rowneruuid": "e61c3e6b-9c54-4b81-8ce5-f9039c1d04d9"
5508             }
5509         }
5510       ],
5511       "responses": {
5512         "400": {
5513           "description": "The request is invalid."
5514         },
5515         "204": {
5516           "description": "The DOXM entry is updated."
5517         }
5518       }
5519     }
5520   }
5521 },
5522 "parameters": {
5523   "interface" : {
5524     "in" : "query",
5525     "name" : "if",
5526     "type" : "string",

```

```

5527     "enum" : ["oic.if.baseline"]
5528   }
5529 },
5530 "definitions": {
5531   "Doxm" : {
5532     "properties": {
5533       "rowneruuid": {
5534         "description": "Format pattern according to IETF RFC 4122.",
5535         "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-
5536 9]{12}$",
5537         "type": "string"
5538       },
5539       "oxms": {
5540         "description": "List of supported owner transfer methods.",
5541         "items": {
5542           "description": "The Device owner transfer methods that may be selected at Device on-
5543 boarding. Each value indicates a specific Owner Transfer method0 - Numeric OTM identifier for the
5544 Just-Works method (oic.sec.doxm.jw)1 - Numeric OTM identifier for the random PIN method
5545 (oic.sec.doxm.rdp)2 - Numeric OTM identifier for the manufacturer certificate method
5546 (oic.sec.doxm.mfgcert)3 - Numeric OTM identifier for the decap method (oic.sec.doxm.dcap)
5547 (deprecated).",
5548         "type": "integer"
5549       },
5550       "readOnly": true,
5551       "type": "array"
5552     },
5553     "devowneruuid": {
5554       "description": "Format pattern according to IETF RFC 4122.",
5555       "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-
5556 9]{12}$",
5557       "type": "string"
5558     },
5559     "deviceuuid": {
5560       "description": "The uuid formatted identity of the Device\nFormat pattern according to
5561 IETF RFC 4122.",
5562       "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-
5563 9]{12}$",
5564       "type": "string"
5565     },
5566     "owned": {
5567       "description": "Ownership status flag.",
5568       "type": "boolean"
5569     },
5570     "n": {
5571       "$ref":
5572 "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
5573 schema.json#/definitions/n"
5574     },
5575     "id": {
5576       "$ref":
5577 "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
5578 schema.json#/definitions/id"
5579     },
5580     "oxmsel": {
5581       "description": "The selected owner transfer method used during on-boarding\nThe Device
5582 owner transfer methods that may be selected at Device on-boarding. Each value indicates a specific
5583 Owner Transfer method0 - Numeric OTM identifier for the Just-Works method (oic.sec.doxm.jw)1 -
5584 Numeric OTM identifier for the random PIN method (oic.sec.doxm.rdp)2 - Numeric OTM identifier for
5585 the manufacturer certificate method (oic.sec.doxm.mfgcert)3 - Numeric OTM identifier for the decap
5586 method (oic.sec.doxm.dcap) (deprecated).",
5587       "type": "integer"
5588     },
5589     "sct": {
5590       "description": "Bitmask encoding of supported credential types\nCredential Types -
5591 Cred type encoded as a bitmask.0 - Empty credential used for testing1 - Symmetric pair-wise key2 -
5592 Symmetric group key4 - Asymmetric signing key8 - Asymmetric signing key with certificatel6 - PIN or
5593 password32 - Asymmetric encryption key.",
5594       "maximum": 63,
5595       "minimum": 0,
5596       "type": "integer",
5597       "readOnly": true

```

```

5598     },
5599     "rt" : {
5600         "description": "Resource Type of the Resource.",
5601         "items": {
5602             "maxLength": 64,
5603             "type": "string",
5604             "enum": ["oic.r.doxm"]
5605         },
5606         "minItems": 1,
5607         "readOnly": true,
5608         "type": "array"
5609     },
5610     "if": {
5611         "description": "The interface set supported by this Resource.",
5612         "items": {
5613             "enum": [
5614                 "oic.if.baseline"
5615             ],
5616             "type": "string"
5617         },
5618         "minItems": 1,
5619         "readOnly": true,
5620         "type": "array"
5621     }
5622 },
5623 "type" : "object",
5624 "required": ["oxms", "oxmsel", "sct", "owned", "deviceuuid", "devowneruuid", "rowneruuid"]
5625 },
5626 "Doxm-Update" : {
5627     "properties": {
5628         "rowneruuid": {
5629             "description": "Format pattern according to IETF RFC 4122.",
5630             "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-
5631 9]{12}$",
5632             "type": "string"
5633         },
5634         "devowneruuid": {
5635             "description": "Format pattern according to IETF RFC 4122.",
5636             "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-
5637 9]{12}$",
5638             "type": "string"
5639         },
5640         "deviceuuid": {
5641             "description": "The uuid formatted identity of the Device\nFormat pattern according to
5642 IETF RFC 4122.",
5643             "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-
5644 9]{12}$",
5645             "type": "string"
5646         },
5647         "owned": {
5648             "description": "Ownership status flag.",
5649             "type": "boolean"
5650         },
5651         "oxmsel": {
5652             "description": "The selected owner transfer method used during on-boarding\nThe Device
5653 owner transfer methods that may be selected at Device on-boarding. Each value indicates a specific
5654 Owner Transfer method0 - Numeric OTM identifier for the Just-Works method (oic.sec.doxm.jw)1 -
5655 Numeric OTM identifier for the random PIN method (oic.sec.doxm.rdp)2 - Numeric OTM identifier for
5656 the manufacturer certificate method (oic.sec.doxm.mfgcert)3 - Numeric OTM identifier for the decap
5657 method (oic.sec.doxm.dcap) (deprecated).",
5658             "type": "integer"
5659         }
5660     },
5661     "type" : "object"
5662 }
5663 }
5664 }
5665

```

5666 **C.5.5 Property definition**

5667 Table C-7 defines the Properties that are part of the "oic.r.doxm" Resource Type.

5668 **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).

5669 **C.5.6 CRUDN behaviour**

5670 Table C-8 defines the CRUDN operations that are supported on the "oic.r.doxm" Resource Type.

5671 **Table C-8 – The CRUDN operations of the Resource with type "rt" = "oic.r.doxm".**

Create	Read	Update	Delete	Notify
	get	post		observe

5672 **C.6 Device Provisioning Status**

5673 **C.6.1 Introduction**

5674 This Resource specifies Device provisioning status.

5675

5676 **C.6.2 Well-known URI**

5677 /oic/sec/pstat

5678 **C.6.3 Resource type**

5679 The Resource Type is defined as: "oic.r.pstat".

5680 **C.6.4 OpenAPI 2.0 definition**

```

5681 {
5682   "swagger": "2.0",
5683   "info": {
5684     "title": "Device Provisioning Status",
5685     "version": "v1.0-20191001",
5686     "license": {
5687       "name": "OCF Data Model License",
5688       "url":
5689         "https://github.com/openconnectivityfoundation/core/blob/e28a9e0a92e17042ba3e83661e4c0fbce8bdc4ba/LI
5690         CENSE.md",
5691       "x-copyright": "copyright 2016-2017, 2019 Open Connectivity Foundation, Inc. All rights
5692       reserved."
5693     },
5694     "termsOfService": "https://openconnectivityfoundation.github.io/core/DISCLAIMER.md"
5695   },
5696   "schemes": ["http"],
5697   "consumes": ["application/json"],
5698   "produces": ["application/json"],
5699   "paths": {
5700     "/oic/sec/pstat" : {
5701       "get": {
5702         "description": "This Resource specifies Device provisioning status.\n",
5703         "parameters": [
5704           {"$ref": "#/parameters/interface"}
5705         ],
5706         "responses": {
5707           "200": {
5708             "description": "",
5709             "x-example":
5710               {
5711                 "rt": ["oic.r.pstat"],
5712                 "dos": {"s": 3, "p": true},
5713                 "isop": true,
5714                 "cm": 8,
5715                 "tm": 60,
5716                 "om": 2,
5717                 "sm": 7,
5718                 "rowneruuid": "de305d54-75b4-431b-adb2-eb6b9e546014"
5719               },
5720             "schema": { "$ref": "#/definitions/Pstat" }
5721           },
5722           "400": {

```

```

5723         "description" : "The request is invalid."
5724     }
5725 }
5726 },
5727 "post": {
5728     "description": "Sets or updates Device provisioning status data.\n",
5729     "parameters": [
5730         {"$ref": "#/parameters/interface"},
5731         {
5732             "name": "body",
5733             "in": "body",
5734             "required": true,
5735             "schema": { "$ref": "#/definitions/Pstat-Update" },
5736             "x-example":
5737                 {
5738                     "dos": {"s": 3},
5739                     "tm": 60,
5740                     "om": 2,
5741                     "rowneruuid": "de305d54-75b4-431b-adb2-eb6b9e546014"
5742                 }
5743         }
5744     ],
5745     "responses": {
5746         "400": {
5747             "description": "The request is invalid."
5748         },
5749         "204": {
5750             "description": "The PSTAT entry is updated."
5751         }
5752     }
5753 }
5754 },
5755 },
5756 "parameters": {
5757     "interface" : {
5758         "in" : "query",
5759         "name" : "if",
5760         "type" : "string",
5761         "enum" : ["oic.if.baseline"]
5762     }
5763 },
5764 "definitions": {
5765     "Pstat" : {
5766         "properties": {
5767             "rowneruuid": {
5768                 "description": "The UUID formatted identity of the Resource owner\nFormat pattern
5769 according to IETF RFC 4122.",
5770                 "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-
5771 9]{12}$",
5772                 "type": "string"
5773             },
5774             "rt": {
5775                 "description": "Resource Type of the Resource.",
5776                 "items": {
5777                     "maxLength": 64,
5778                     "type": "string",
5779                     "enum": ["oic.r.pstat"]
5780                 },
5781                 "minItems": 1,
5782                 "readOnly": true,
5783                 "type": "array"
5784             },
5785             "om": {
5786                 "description": "Current operational mode\nDevice provisioning operation may be server
5787 directed or client (aka provisioning service) directed. The value is a bitmask encoded as integer
5788 and indicates the provisioning operation modes1 - Server-directed utilizing multiple provisioning
5789 services2 - Server-directed utilizing a single provisioning service4 - Client-directed provisioning8
5790 - Unused16 - Unused32 - Unused64 - Unused128 - Unused.",
5791                 "maximum": 7,
5792                 "minimum": 1,
5793                 "type": "integer"

```

```

5794     },
5795     "cm": {
5796       "description": "Current Device provisioning mode\nDevice provisioning mode maintains a
5797       bitmask of the possible provisioning states of a Device. The value can be either 8 or 16 character
5798       in length. If its only 8 characters it represents the lower byte value1 - Manufacturer reset state2
5799       - Device pairing and owner transfer state4 - Unused8 - Provisioning of credential management
5800       services16 - Provisioning of access management services32 - Provisioning of local ACLs64 - Initiate
5801       Software Version Validation128 - Initiate Secure Software Update.",
5802       "maximum": 255,
5803       "minimum": 0,
5804       "type": "integer",
5805       "readOnly": true
5806     },
5807     "n": {
5808       "$ref":
5809       "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
5810       schema.json#/definitions/n"
5811     },
5812     "id": {
5813       "$ref":
5814       "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
5815       schema.json#/definitions/id"
5816     },
5817     "isop": {
5818       "description": "true indicates Device is operational.",
5819       "readOnly": true,
5820       "type": "boolean"
5821     },
5822     "tm": {
5823       "description": "Target Device provisioning mode\nDevice provisioning mode maintains a
5824       bitmask of the possible provisioning states of a Device. The value can be either 8 or 16 character
5825       in length. If its only 8 characters it represents the lower byte value1 - Manufacturer reset state2
5826       - Device pairing and owner transfer state4 - Unused8 - Provisioning of credential management
5827       services16 - Provisioning of access management services32 - Provisioning of local ACLs64 - Initiate
5828       Software Version Validation128 - Initiate Secure Software Update.",
5829       "maximum": 255,
5830       "minimum": 0,
5831       "type": "integer"
5832     },
5833     "sm": {
5834       "description": "Supported operational modes\nDevice provisioning operation may be server
5835       directed or client (aka provisioning service) directed. The value is a bitmask encoded as integer
5836       and indicates the provisioning operation modes1 - Server-directed utilizing multiple provisioning
5837       services2 - Server-directed utilizing a single provisioning service4 - Client-directed provisioning8
5838       - Unused16 - Unused32 - Unused64 - Unused128 - Unused.",
5839       "maximum": 7,
5840       "minimum": 1,
5841       "type": "integer",
5842       "readOnly": true
5843     },
5844     "dos": {
5845       "description": "Device on-boarding state\nDevice operation state machine.",
5846       "properties": {
5847         "p": {
5848           "default": true,
5849           "description": "'p' is TRUE when the 's' state is pending until all necessary changes
5850           to Device Resources are complete.",
5851           "readOnly": true,
5852           "type": "boolean"
5853         },
5854         "s": {
5855           "description": "The current or pending operational state.",
5856           "x-detail-desc": [
5857             "0 - RESET - Device reset state.",
5858             "1 - RFOFM - Ready for Device owner transfer method state.",
5859             "2 - RFPPO - Ready for Device provisioning state.",
5860             "3 - RFNOP - Ready for Device normal operation state.",
5861             "4 - SRESET - The Device is in a soft reset state."
5862           ],
5863           "maximum": 4,
5864           "minimum": 0,

```

```

5865         "type": "integer"
5866     }
5867 },
5868 "required": [
5869     "s"
5870 ],
5871 "type": "object"
5872 },
5873 "if" : {
5874     "description": "The interface set supported by this Resource.",
5875     "items": {
5876         "enum": [
5877             "oic.if.baseline"
5878         ],
5879         "type": "string"
5880     },
5881     "minItems": 1,
5882     "readOnly": true,
5883     "type": "array"
5884 }
5885 },
5886 "type" : "object",
5887 "required": ["dos", "isop", "cm", "tm", "om", "sm", "rowneruuid"]
5888 },
5889 "Pstat-Update" : {
5890     "properties": {
5891         "rowneruuid": {
5892             "description": "The UUID formatted identity of the Resource owner\nFormat pattern
5893 according to IETF RFC 4122.",
5894             "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-
5895 9]{12}$",
5896             "type": "string"
5897         },
5898         "om": {
5899             "description": "Current operational mode\nDevice provisioning operation may be server
5900 directed or client (aka provisioning service) directed. The value is a bitmask encoded as integer
5901 and indicates the provisioning operation modes1 - Server-directed utilizing multiple provisioning
5902 services2 - Server-directed utilizing a single provisioning service4 - Client-directed provisioning8
5903 - Unused16 - Unused32 - Unused64 - Unused128 - Unused.",
5904             "maximum": 7,
5905             "minimum": 1,
5906             "type": "integer"
5907         },
5908         "tm": {
5909             "description": "Target Device provisioning mode\nDevice provisioning mode maintains a
5910 bitmask of the possible provisioning states of a Device. The value can be either 8 or 16 character
5911 in length. If its only 8 characters it represents the lower byte value1 - Manufacturer reset state2
5912 - Device pairing and owner transfer state4 - Unused8 - Provisioning of credential management
5913 services16 - Provisioning of access management services32 - Provisioning of local ACLs64 - Initiate
5914 Software Version Validation128 - Initiate Secure Software Update.",
5915             "maximum": 255,
5916             "minimum": 0,
5917             "type": "integer"
5918         },
5919         "dos": {
5920             "description": "Device on-boarding state\nDevice operation state machine.",
5921             "properties": {
5922                 "p": {
5923                     "default": true,
5924                     "description": "'p' is TRUE when the 's' state is pending until all necessary changes
5925 to Device Resources are complete.",
5926                     "readOnly": true,
5927                     "type": "boolean"
5928                 },
5929                 "s": {
5930                     "description": "The current or pending operational state.",
5931                     "x-detail-desc": [
5932                         "0 - RESET - Device reset state.",
5933                         "1 - RFOFM - Ready for Device owner transfer method state.",
5934                         "2 - RFPPO - Ready for Device provisioning state.",
5935                         "3 - RFNOP - Ready for Device normal operation state.",

```

```

5936         "4 - SRESET - The Device is in a soft reset state."
5937     ],
5938     "maximum": 4,
5939     "minimum": 0,
5940     "type": "integer"
5941   }
5942 },
5943   "required": [
5944     "s"
5945   ],
5946   "type": "object"
5947 }
5948 },
5949 "type" : "object"
5950 }
5951 }
5952 }
5953

```

5954 C.6.5 Property definition

5955 Table C-9 defines the Properties that are part of the "oic.r.pstat" Resource Type.

5956 **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.

5957 **C.6.6 CRUDN behaviour**

5958 Table C-10 defines the CRUDN operations that are supported on the "oic.r.pstat" Resource Type.

5959

Table C-10 – The CRUDN operations of the Resource with type "rt" = "oic.r.pstat".

Create	Read	Update	Delete	Notify
	get	post		observe

5960

C.7 Asserted Roles

5961

C.7.1 Introduction

5962

This Resource specifies roles that have been asserted.

5963

5964

C.7.2 Well-known URI

5965

/oic/sec/roles

5966

C.7.3 Resource type

5967

The Resource Type is defined as: "oic.r.roles".

5968

C.7.4 OpenAPI 2.0 definition

5969

```

5970 {
5971   "swagger": "2.0",
5972   "info": {
5973     "title": "Asserted Roles",
5974     "version": "v1.0-20170323",
5975     "license": {
5976       "name": "OCF Data Model License",
5977       "url":
5978         "https://github.com/openconnectivityfoundation/core/blob/e28a9e0a92e17042ba3e83661e4c0fbce8bdc4ba/LI
5979         CENSE.md",
5980       "x-copyright": "copyright 2016-2017, 2019 Open Connectivity Foundation, Inc. All rights
5981       reserved."
5982     },
5983     "termsOfService": "https://openconnectivityfoundation.github.io/core/DISCLAIMER.md"
5984   },
5985   "schemes": ["http"],
5986   "consumes": ["application/json"],
5987   "produces": ["application/json"],
5988   "paths": {
5989     "/oic/sec/roles" : {
5990       "get": {
5991         "description": "This Resource specifies roles that have been asserted.\n",
5992         "parameters": [
5993           {"$ref": "#/parameters/interface"}
5994         ],
5995         "responses": {
5996           "200": {
5997             "description": "",
5998             "x-example":
5999               {
6000                 "roles" :[
6001                   {
6002                     "credid":1,
6003                     "credtype":8,
6004                     "subjectuuid":"00000000-0000-0000-0000-000000000000",
6005                     "publicdata":
6006                       {
6007                         "encoding":"oic.sec.encoding.pem",
6008                         "data":"PEMENCODEDROLECERT"
6009                       },
6010                     "optionaldata":
6011                       {
6012                         "revstat": false,
6013                         "encoding":"oic.sec.encoding.pem",
6014                         "data":"PEMENCODEDISSUERCERT"
6015                       }
6016                   }
6017                 ]
6018             }
6019           }
6020         }
6021       }
6022     }
6023   }
6024 }

```

```

6016         {
6017             "credid":2,
6018             "credtype":8,
6019             "subjectuuiid":"00000000-0000-0000-0000-000000000000",
6020             "publicdata":
6021                 {
6022                     "encoding":"oic.sec.encoding.pem",
6023                     "data":"PEMENCODEDROLECERT"
6024                 },
6025             "optionaldata":
6026                 {
6027                     "revstat": false,
6028                     "encoding":"oic.sec.encoding.pem",
6029                     "data":"PEMENCODEDISSUERCERT"
6030                 }
6031         },
6032     ],
6033     "rt":["oic.r.roles"],
6034     "if":["oic.if.baseline"]
6035 }
6036 ,
6037 "schema": { "$ref": "#/definitions/Roles" }
6038 },
6039 "400": {
6040     "description": "The request is invalid."
6041 }
6042 },
6043 },
6044 "post": {
6045     "description": "Update the roles Resource, i.e., assert new roles to this server.\n\nNew
6046 role certificates that match an existing certificate (i.e., publicdata\nand optionaldata are the
6047 same) are not added to the Resource (and 204 is\nreturned).\n\nThe provided credid values are
6048 ignored, the Resource assigns its own.\n",
6049     "parameters": [
6050         { "$ref": "#/parameters/interface" },
6051         {
6052             "name": "body",
6053             "in": "body",
6054             "required": true,
6055             "schema": { "$ref": "#/definitions/Roles-update" },
6056             "x-example":
6057                 {
6058                     "roles" :[
6059                         {
6060                             "credid":1,
6061                             "credtype":8,
6062                             "subjectuuiid":"00000000-0000-0000-0000-000000000000",
6063                             "publicdata":
6064                                 {
6065                                     "encoding":"oic.sec.encoding.pem",
6066                                     "data":"PEMENCODEDROLECERT"
6067                                 },
6068                             "optionaldata":
6069                                 {
6070                                     "revstat": false,
6071                                     "encoding":"oic.sec.encoding.pem",
6072                                     "data":"PEMENCODEDISSUERCERT"
6073                                 }
6074                         },
6075                         {
6076                             "credid":2,
6077                             "credtype":8,
6078                             "subjectuuiid":"00000000-0000-0000-0000-000000000000",
6079                             "publicdata":
6080                                 {
6081                                     "encoding":"oic.sec.encoding.pem",
6082                                     "data":"PEMENCODEDROLECERT"
6083                                 },
6084                             "optionaldata":
6085                                 {
6086                                     "revstat": false,

```

```

6087         "encoding": "oic.sec.encoding.pem",
6088         "data": "PEMENCODEDISSUERCERT"
6089     }
6090 }
6091 ]
6092 }
6093 }
6094 ],
6095 "responses": {
6096     "400": {
6097         "description": "The request is invalid."
6098     },
6099     "204": {
6100         "description": "The roles entry is updated."
6101     }
6102 }
6103 },
6104 "delete": {
6105     "description": "Deletes roles Resource entries.\nWhen DELETE is used without query
6106 parameters, all the roles entries are deleted.\nWhen DELETE is used with a query parameter, only the
6107 entries matching\nthe query parameter are deleted.\n",
6108     "parameters": [
6109         {"$ref": "#/parameters/interface"},
6110         {"$ref": "#/parameters/roles-filtered"}
6111     ],
6112     "responses": {
6113         "200": {
6114             "description": "The specified or all roles Resource entries have been successfully
6115 deleted."
6116         },
6117         "400": {
6118             "description": "The request is invalid."
6119         }
6120     }
6121 }
6122 }
6123 },
6124 "parameters": {
6125     "interface": {
6126         "in": "query",
6127         "name": "if",
6128         "type": "string",
6129         "enum": ["oic.if.baseline"]
6130     },
6131     "roles-filtered": {
6132         "in": "query",
6133         "name": "credid",
6134         "required": false,
6135         "type": "integer",
6136         "description": "Only applies to the credential with the specified credid.",
6137         "x-example": 2112
6138     }
6139 },
6140 "definitions": {
6141     "Roles": {
6142         "properties": {
6143             "rt": {
6144                 "description": "Resource Type of the Resource.",
6145                 "items": {
6146                     "maxLength": 64,
6147                     "type": "string",
6148                     "enum": ["oic.r.roles"]
6149                 },
6150                 "minItems": 1,
6151                 "readOnly": true,
6152                 "type": "array"
6153             },
6154             "n": {
6155                 "$ref":
6156 "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
6157 schema.json#/definitions/n"

```

```

6158     },
6159     "id": {
6160         "$ref":
6161         "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
6162         schema.json#/definitions/id"
6163     },
6164     "roles": {
6165         "description": "List of role certificates.",
6166         "items": {
6167             "properties": {
6168                 "credid": {
6169                     "description": "Local reference to a credential Resource.",
6170                     "type": "integer"
6171                 },
6172                 "credtype": {
6173                     "description": "Representation of this credential's type\nCredential Types - Cred
6174 type encoded as a bitmask.0 - Empty credential used for testing1 - Symmetric pair-wise key2 -
6175 Symmetric group key4 - Asymmetric signing key8 - Asymmetric signing key with certificatel6 - PIN or
6176 password32 - Asymmetric encryption key.",
6177                     "maximum": 63,
6178                     "minimum": 0,
6179                     "type": "integer"
6180                 },
6181                 "credusage": {
6182                     "description": "A string that provides hints about how/where the cred is used\nThe
6183 type of credusage.oic.sec.cred.trustca - Trust certificateoic.sec.cred.cert -
6184 Certificateoic.sec.cred.rolecert - Role Certificateoic.sec.cred.mfgtrustca - Manufacturer
6185 Certificate Trust Anchoroic.sec.cred.mfgcert - Manufacturer Certificate.",
6186                     "enum": [
6187                         "oic.sec.cred.trustca",
6188                         "oic.sec.cred.cert",
6189                         "oic.sec.cred.rolecert",
6190                         "oic.sec.cred.mfgtrustca",
6191                         "oic.sec.cred.mfgcert"
6192                     ],
6193                     "type": "string"
6194                 },
6195                 "crms": {
6196                     "description": "The refresh methods that may be used to update this credential.",
6197                     "items": {
6198                         "description": "Each enum represents a method by which the credentials are
6199 refreshed.oic.sec.crm.pro - Credentials refreshed by a provisioning serviceoic.sec.crm.rdp -
6200 Credentials refreshed by a key agreement protocol and random PINoic.sec.crm.psk - Credentials
6201 refreshed by a key agreement protocoloic.sec.crm.skdc - Credentials refreshed by a key distribution
6202 serviceoic.sec.crm.pk10 - Credentials refreshed by a PKCS#10 request to a CA.",
6203                         "enum": [
6204                             "oic.sec.crm.pro",
6205                             "oic.sec.crm.psk",
6206                             "oic.sec.crm.rdp",
6207                             "oic.sec.crm.skdc",
6208                             "oic.sec.crm.pk10"
6209                         ],
6210                         "type": "string"
6211                     },
6212                     "type": "array"
6213                 },
6214                 "optionaldata": {
6215                     "description": "Credential revocation status information\nOptional credential
6216 contents describes revocation status for this credential.",
6217                     "properties": {
6218                         "data": {
6219                             "description": "This is the encoded structure.",
6220                             "type": "string"
6221                         },
6222                         "encoding": {
6223                             "description": "A string specifying the encoding format of the data contained in
6224 the optdata.",
6225                             "x-detail-desc": [
6226                                 "oic.sec.encoding.jwt - RFC7517 JSON web token (JWT) encoding.",
6227                                 "oic.sec.encoding.cwt - RFC CBOR web token (CWT) encoding.",
6228                                 "oic.sec.encoding.base64 - Base64 encoded object.",

```

```

6229         "oic.sec.encoding.pem - Encoding for PEM encoded certificate or chain.",
6230         "oic.sec.encoding.der - Encoding for DER encoded certificate.",
6231         "oic.sec.encoding.raw - Raw hex encoded data."
6232     ],
6233     "enum": [
6234         "oic.sec.encoding.jwt",
6235         "oic.sec.encoding.cwt",
6236         "oic.sec.encoding.base64",
6237         "oic.sec.encoding.pem",
6238         "oic.sec.encoding.der",
6239         "oic.sec.encoding.raw"
6240     ],
6241     "type": "string"
6242 },
6243 "revstat": {
6244     "description": "Revocation status flag - true = revoked.",
6245     "type": "boolean"
6246 },
6247 },
6248 "required": [
6249     "revstat"
6250 ],
6251 "type": "object"
6252 },
6253 "period": {
6254     "description": "String with RFC5545 Period.",
6255     "type": "string"
6256 },
6257 "privatedata": {
6258     "description": "Private credential information\nCredential Resource non-public
6259 contents.",
6260     "properties": {
6261         "data": {
6262             "description": "The encoded value.",
6263             "maxLength": 3072,
6264             "type": "string"
6265         },
6266         "encoding": {
6267             "description": "A string specifying the encoding format of the data contained in
6268 the privdata.",
6269             "x-detail-desc": [
6270                 "oic.sec.encoding.jwt - RFC7517 JSON web token (JWT) encoding.",
6271                 "oic.sec.encoding.cwt - RFC CBOR web token (CWT) encoding.",
6272                 "oic.sec.encoding.base64 - Base64 encoded object.",
6273                 "oic.sec.encoding.uri - URI reference.",
6274                 "oic.sec.encoding.handle - Data is contained in a storage sub-system
6275 referenced using a handle.",
6276                 "oic.sec.encoding.raw - Raw hex encoded data."
6277             ],
6278             "enum": [
6279                 "oic.sec.encoding.jwt",
6280                 "oic.sec.encoding.cwt",
6281                 "oic.sec.encoding.base64",
6282                 "oic.sec.encoding.uri",
6283                 "oic.sec.encoding.handle",
6284                 "oic.sec.encoding.raw"
6285             ],
6286             "type": "string"
6287         },
6288         "handle": {
6289             "description": "Handle to a key storage Resource.",
6290             "type": "integer"
6291         }
6292     },
6293     "required": [
6294         "encoding"
6295     ],
6296     "type": "object"
6297 },
6298 "publicdata": {
6299     "description": "Public credential information.",

```

```

6300         "properties": {
6301             "data": {
6302                 "description": "This is the encoded value.",
6303                 "maxLength": 3072,
6304                 "type": "string"
6305             },
6306             "encoding": {
6307                 "description": "A string specifying the encoding format of the data contained in
6308 the pubdata.",
6309                 "x-detail-desc": [
6310                     "oic.sec.encoding.jwt - RFC7517 JSON web token (JWT) encoding.",
6311                     "oic.sec.encoding.cwt - RFC CBOR web token (CWT) encoding.",
6312                     "oic.sec.encoding.base64 - Base64 encoded object.",
6313                     "oic.sec.encoding.uri - URI reference.",
6314                     "oic.sec.encoding.pem - Encoding for PEM encoded certificate or chain.",
6315                     "oic.sec.encoding.der - Encoding for DER encoded certificate.",
6316                     "oic.sec.encoding.raw - Raw hex encoded data."
6317                 ],
6318                 "enum": [
6319                     "oic.sec.encoding.jwt",
6320                     "oic.sec.encoding.cwt",
6321                     "oic.sec.encoding.base64",
6322                     "oic.sec.encoding.uri",
6323                     "oic.sec.encoding.pem",
6324                     "oic.sec.encoding.der",
6325                     "oic.sec.encoding.raw"
6326                 ],
6327                 "type": "string"
6328             }
6329         },
6330         "type": "object"
6331     },
6332     "roleid": {
6333         "description": "The role this credential possesses\nSecurity role specified as an
6334 <Authority> & <Rolename>. A NULL <Authority> refers to the local entity or Device.",
6335         "properties": {
6336             "authority": {
6337                 "description": "The Authority component of the entity being identified. A NULL
6338 <Authority> refers to the local entity or Device.",
6339                 "type": "string"
6340             },
6341             "role": {
6342                 "description": "The ID of the role being identified.",
6343                 "type": "string"
6344             }
6345         },
6346         "required": [
6347             "role"
6348         ],
6349         "type": "object"
6350     },
6351     "subjectuuid": {
6352         "anyOf": [
6353             {
6354                 "description": "The id of the Device, which the cred entry applies to or \"*\
6355 for wildcard identity.",
6356                 "pattern": "^\\*$",
6357                 "type": "string"
6358             },
6359             {
6360                 "description": "Format pattern according to IETF RFC 4122.",
6361                 "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-
6362 F0-9]{12}$",
6363                 "type": "string"
6364             }
6365         ]
6366     }
6367 },
6368 "type": "object"
6369 },
6370 "type": "array"

```

```

6371     },
6372     "if": {
6373         "description": "The interface set supported by this Resource.",
6374         "items": {
6375             "enum": [
6376                 "oic.if.baseline"
6377             ],
6378             "type": "string"
6379         },
6380         "minItems": 1,
6381         "readOnly": true,
6382         "type": "array"
6383     }
6384 },
6385 "type": "object",
6386 "required": ["roles"]
6387 },
6388 "Roles-update" : {
6389     "properties": {
6390         "roles": {
6391             "description": "List of role certificates.",
6392             "items": {
6393                 "properties": {
6394                     "credid": {
6395                         "description": "Local reference to a credential Resource.",
6396                         "type": "integer"
6397                     },
6398                     "credtype": {
6399                         "description": "Representation of this credential's type\nCredential Types - Cred
6400 type encoded as a bitmask.0 - Empty credential used for testing1 - Symmetric pair-wise key2 -
6401 Symmetric group key4 - Asymmetric signing key8 - Asymmetric signing key with certificatel6 - PIN or
6402 password32 - Asymmetric encryption key.",
6403                         "maximum": 63,
6404                         "minimum": 0,
6405                         "type": "integer"
6406                     },
6407                     "credusage": {
6408                         "description": "A string that provides hints about how/where the cred is used\nThe
6409 type of credusage.oic.sec.cred.trustca - Trust certificateoic.sec.cred.cert -
6410 Certificateoic.sec.cred.rolecert - Role Certificateoic.sec.cred.mfgtrustca - Manufacturer
6411 Certificate Trust Anchoroic.sec.cred.mfgcert - Manufacturer Certificate.",
6412                         "enum": [
6413                             "oic.sec.cred.trustca",
6414                             "oic.sec.cred.cert",
6415                             "oic.sec.cred.rolecert",
6416                             "oic.sec.cred.mfgtrustca",
6417                             "oic.sec.cred.mfgcert"
6418                         ],
6419                         "type": "string"
6420                     },
6421                     "crms": {
6422                         "description": "The refresh methods that may be used to update this credential.",
6423                         "items": {
6424                             "description": "Each enum represents a method by which the credentials are
6425 refreshed.oic.sec.crm.pro - Credentials refreshed by a provisioning serviceoic.sec.crm.rdp -
6426 Credentials refreshed by a key agreement protocol and random PINoic.sec.crm.psk - Credentials
6427 refreshed by a key agreement protocoloic.sec.crm.skdc - Credentials refreshed by a key distribution
6428 serviceoic.sec.crm.pk10 - Credentials refreshed by a PKCS#10 request to a CA.",
6429                             "enum": [
6430                                 "oic.sec.crm.pro",
6431                                 "oic.sec.crm.psk",
6432                                 "oic.sec.crm.rdp",
6433                                 "oic.sec.crm.skdc",
6434                                 "oic.sec.crm.pk10"
6435                             ],
6436                             "type": "string"
6437                         },
6438                         "type": "array"
6439                     },
6440                     "optionaldata": {
6441                         "description": "Credential revocation status information\nOptional credential

```



```

6442 contents describes revocation status for this credential.",
6443     "properties": {
6444         "data": {
6445             "description": "This is the encoded structure.",
6446             "type": "string"
6447         },
6448         "encoding": {
6449             "description": "A string specifying the encoding format of the data contained in
6450 the optdata.",
6451             "x-detail-desc": [
6452                 "oic.sec.encoding.jwt - RFC7517 JSON web token (JWT) encoding.",
6453                 "oic.sec.encoding.cwt - RFC CBOR web token (CWT) encoding.",
6454                 "oic.sec.encoding.base64 - Base64 encoded object.",
6455                 "oic.sec.encoding.pem - Encoding for PEM encoded certificate or chain.",
6456                 "oic.sec.encoding.der - Encoding for DER encoded certificate.",
6457                 "oic.sec.encoding.raw - Raw hex encoded data."
6458             ],
6459             "enum": [
6460                 "oic.sec.encoding.jwt",
6461                 "oic.sec.encoding.cwt",
6462                 "oic.sec.encoding.base64",
6463                 "oic.sec.encoding.pem",
6464                 "oic.sec.encoding.der",
6465                 "oic.sec.encoding.raw"
6466             ],
6467             "type": "string"
6468         },
6469         "revstat": {
6470             "description": "Revocation status flag - true = revoked.",
6471             "type": "boolean"
6472         }
6473     },
6474     "required": [
6475         "revstat"
6476     ],
6477     "type": "object"
6478 },
6479 "period": {
6480     "description": "String with RFC5545 Period.",
6481     "type": "string"
6482 },
6483 "privatedata": {
6484     "description": "Private credential information\nCredential Resource non-public
6485 contents.",
6486     "properties": {
6487         "data": {
6488             "description": "The encoded value.",
6489             "maxLength": 3072,
6490             "type": "string"
6491         },
6492         "encoding": {
6493             "description": "A string specifying the encoding format of the data contained in
6494 the privdata.",
6495             "x-detail-desc": [
6496                 "oic.sec.encoding.jwt - RFC7517 JSON web token (JWT) encoding.",
6497                 "oic.sec.encoding.cwt - RFC CBOR web token (CWT) encoding.",
6498                 "oic.sec.encoding.base64 - Base64 encoded object.",
6499                 "oic.sec.encoding.uri - URI reference.",
6500                 "oic.sec.encoding.handle - Data is contained in a storage sub-system
6501 referenced using a handle.",
6502                 "oic.sec.encoding.raw - Raw hex encoded data."
6503             ],
6504             "enum": [
6505                 "oic.sec.encoding.jwt",
6506                 "oic.sec.encoding.cwt",
6507                 "oic.sec.encoding.base64",
6508                 "oic.sec.encoding.uri",
6509                 "oic.sec.encoding.handle",
6510                 "oic.sec.encoding.raw"
6511             ],
6512             "type": "string"

```

```

6513         },
6514         "handle": {
6515             "description": "Handle to a key storage Resource.",
6516             "type": "integer"
6517         }
6518     },
6519     "required": [
6520         "encoding"
6521     ],
6522     "type": "object"
6523 },
6524 "publicdata": {
6525     "description": "Public credential information.",
6526     "properties": {
6527         "data": {
6528             "description": "The encoded value.",
6529             "maxLength": 3072,
6530             "type": "string"
6531         },
6532         "encoding": {
6533             "description": "A string specifying the encoding format of the data contained in
6534 the pubdata.",
6535             "x-detail-desc": [
6536                 "oic.sec.encoding.jwt - RFC7517 JSON web token (JWT) encoding.",
6537                 "oic.sec.encoding.cwt - RFC CBOR web token (CWT) encoding.",
6538                 "oic.sec.encoding.base64 - Base64 encoded object.",
6539                 "oic.sec.encoding.uri - URI reference.",
6540                 "oic.sec.encoding.pem - Encoding for PEM encoded certificate or chain.",
6541                 "oic.sec.encoding.der - Encoding for DER encoded certificate.",
6542                 "oic.sec.encoding.raw - Raw hex encoded data."
6543             ],
6544             "enum": [
6545                 "oic.sec.encoding.jwt",
6546                 "oic.sec.encoding.cwt",
6547                 "oic.sec.encoding.base64",
6548                 "oic.sec.encoding.uri",
6549                 "oic.sec.encoding.pem",
6550                 "oic.sec.encoding.der",
6551                 "oic.sec.encoding.raw"
6552             ],
6553             "type": "string"
6554         }
6555     },
6556     "type": "object"
6557 },
6558 "roleid": {
6559     "description": "The role this credential possesses\nSecurity role specified as an
6560 <Authority> & <Rolename>. A NULL <Authority> refers to the local entity or Device.",
6561     "properties": {
6562         "authority": {
6563             "description": "The Authority component of the entity being identified. A NULL
6564 <Authority> refers to the local entity or Device.",
6565             "type": "string"
6566         },
6567         "role": {
6568             "description": "The ID of the role being identified.",
6569             "type": "string"
6570         }
6571     },
6572     "required": [
6573         "role"
6574     ],
6575     "type": "object"
6576 },
6577 "subjectuuid": {
6578     "anyOf": [
6579         {
6580             "description": "The id of the Device, which the cred entry applies to or \"*\n"
6581 for wildcard identity.",
6582             "pattern": "^\\*$",
6583             "type": "string"

```

```

6584         },
6585         {
6586             "description": "Format pattern according to IETF RFC 4122.",
6587             "pattern": "^[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-
6588 F0-9]{12}$",
6589             "type": "string"
6590         }
6591     ]
6592 },
6593 },
6594     "type": "object"
6595 },
6596     "type": "array"
6597 }
6598 },
6599     "type": "object",
6600     "required": ["roles"]
6601 }
6602 }
6603 }
6604

```

6605 C.7.5 Property definition

6606 Table C-11 defines the Properties that are part of the "oic.r.roles" Resource Type.

6607 **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.

6608 C.7.6 CRUDN behaviour

6609 Table C-12 defines the CRUDN operations that are supported on the "oic.r.roles" Resource Type.

6610 **Table C-12 – The CRUDN operations of the Resource with type "rt" = "oic.r.roles".**

Create	Read	Update	Delete	Notify
	get	post	delete	observe

6611 C.8 Security Profile

6612 C.8.1 Introduction

6613 Resource specifying supported and active security profile(s).

6614

6615 C.8.2 Well-known URI

6616 /oic/sec/sp

6617 C.8.3 Resource type

6618 The Resource Type is defined as: "oic.r.sp".

6619 C.8.4 OpenAPI 2.0 definition

```
6620 {
6621   "swagger": "2.0",
6622   "info": {
6623     "title": "Security Profile",
6624     "version": "v1.0-20190208",
6625     "license": {
6626       "name": "OCF Data Model License",
6627       "url":
6628         "https://github.com/openconnectivityfoundation/core/blob/e28a9e0a92e17042ba3e83661e4c0fbce8bdc4ba/LI
6629         CENSE.md",
6630       "x-copyright": "copyright 2016-2017, 2019 Open Connectivity Foundation, Inc. All rights
6631 reserved."
6632     },
6633     "termsOfService": "https://openconnectivityfoundation.github.io/core/DISCLAIMER.md"
6634   },
6635   "schemes": ["http"],
6636   "consumes": ["application/json"],
6637   "produces": ["application/json"],
6638   "paths": {
6639     "/oic/sec/sp" : {
6640       "get": {
6641         "description": "Resource specifying supported and active security profile(s).\n",
6642         "parameters": [
6643           { "$ref": "#/parameters/interface" }
6644         ],
6645         "responses": {
6646           "200": {
6647             "description": "",
6648             "x-example":
6649               {
6650                 "rt": ["oic.r.sp"],
6651                 "supportedprofiles" : ["1.3.6.1.4.1.51414.0.0.1.0", " 1.3.6.1.4.1.51414.0.0.2.0"],
6652                 "currentprofile" : "1.3.6.1.4.1.51414.0.0.1.0"
6653               },
6654             "schema": { "$ref": "#/definitions/SP" }
6655           },
6656           "400": {
6657             "description": "The request is invalid."
6658           }
6659         }
6660       },
6661       "post": {
6662         "description": "Sets or updates Device provisioning status data.\n",
6663         "parameters": [
6664           { "$ref": "#/parameters/interface" },
6665           {
6666             "name": "body",
6667             "in": "body",
6668             "required": true,
6669             "schema": { "$ref": "#/definitions/SP-Update" },
6670             "x-example":
6671               {
6672                 "supportedprofiles" : ["1.3.6.1.4.1.51414.0.0.1.0", " 1.3.6.1.4.1.51414.0.0.2.0"],
6673                 "currentprofile" : "1.3.6.1.4.1.51414.0.0.1.0"
6674               }
6675           }
6676         ],
6677         "responses": {
6678           "200": {
6679             "description": "",
6680             "x-example":
6681               {
6682                 "rt": ["oic.r.sp"],
6683                 "supportedprofiles" : ["1.3.6.1.4.1.51414.0.0.1.0", " 1.3.6.1.4.1.51414.0.0.2.0"],
6684                 "currentprofile" : "1.3.6.1.4.1.51414.0.0.1.0"
6685               }
6686         }
6687       }
6688     }
6689   }
6690 }
```

```

6685         },
6686         "schema": { "$ref": "#/definitions/SP" }
6687     },
6688     "400": {
6689         "description": "The request is invalid."
6690     }
6691 }
6692 }
6693 }
6694 },
6695 "parameters": {
6696     "interface" : {
6697         "in" : "query",
6698         "name" : "if",
6699         "type" : "string",
6700         "enum" : ["oic.if.baseline"]
6701     }
6702 },
6703 "definitions": {
6704     "SP" : {
6705         "properties": {
6706             "rt": {
6707                 "description": "Resource Type of the Resource.",
6708                 "items": {
6709                     "maxLength": 64,
6710                     "type": "string",
6711                     "enum": ["oic.r.sp"]
6712                 },
6713                 "minItems": 1,
6714                 "readOnly": true,
6715                 "type": "array"
6716             },
6717             "n": {
6718                 "$ref":
6719 "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
6720 schema.json#/definitions/n"
6721             },
6722             "id": {
6723                 "$ref":
6724 "https://openconnectivityfoundation.github.io/core/schemas/oic.common.properties.core-
6725 schema.json#/definitions/id"
6726             },
6727             "currentprofile": {
6728                 "description": "Security Profile currently active.",
6729                 "type": "string"
6730             },
6731             "supportedprofiles": {
6732                 "description": "Array of supported Security Profiles.",
6733                 "items": {
6734                     "type": "string"
6735                 },
6736                 "type": "array"
6737             },
6738             "if": {
6739                 "description": "The interface set supported by this Resource.",
6740                 "items": {
6741                     "enum": [
6742                         "oic.if.baseline"
6743                     ],
6744                     "type": "string"
6745                 },
6746                 "minItems": 1,
6747                 "readOnly": true,
6748                 "type": "array"
6749             }
6750         },
6751         "type" : "object",
6752         "required": ["supportedprofiles", "currentprofile"]
6753     },
6754     "SP-Update" : {
6755         "properties": {

```

```

6756     "currentprofile": {
6757         "description": "Security Profile currently active.",
6758         "type": "string"
6759     },
6760     "supportedprofiles": {
6761         "description": "Array of supported Security Profiles.",
6762         "items": {
6763             "type": "string"
6764         },
6765         "type": "array"
6766     }
6767 },
6768 "type" : "object"
6769 }
6770 }
6771 }
6772

```

6773 **C.8.5 Property definition**

6774 Table C-13 defines the Properties that are part of the "oic.r.sp" Resource Type.

6775 **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.

6776 **C.8.6 CRUDN behaviour**

6777 Table C-14 defines the CRUDN operations that are supported on the "oic.r.sp" Resource Type.

6778 **Table C-14 – The CRUDN operations of the Resource with type "rt" = "oic.r.sp".**

Create	Read	Update	Delete	Notify
	get	post		observe

6779

6780
6781
6782
6783

Annex D (informative)

OID definitions

6784 This annex captures the OIDs defined throughout the document. The OIDs listed are intended to
6785 be used within the context of an X.509 v3 certificate. MAX is an upper bound for SEQUENCES of
6786 UTF8Strings and OBJECT IDENTIFIERS and should not exceed 255.

```
6787 id-OCF OBJECT IDENTIFIER ::= { iso(1) identified-organization(3) dod(6) internet(1)
6788     private(4) enterprise(1) OCF(51414) }
6789
6790 -- OCF Security specific OIDs
6791
6792 id-ocfSecurity OBJECT IDENTIFIER ::= { id-OCF 0 }
6793 id-ocfX509Extensions OBJECT IDENTIFIER ::= { id-OCF 1 }
6794
6795 -- OCF Security Categories
6796
6797 id-ocfSecurityProfile ::= { id-ocfSecurity 0 }
6798 id-ocfCertificatePolicy ::= { id-ocfSecurity 1 }
6799
6800 -- OCF Security Profiles
6801
6802 sp-unspecified ::= OBJECT IDENTIFIER { id-ocfSecurityProfile 0 }
6803 sp-baseline ::= OBJECT IDENTIFIER { id-ocfSecurityProfile 1 }
6804 sp-black ::= OBJECT IDENTIFIER { id-ocfSecurityProfile 2 }
6805 sp-blue ::= OBJECT IDENTIFIER { id-ocfSecurityProfile 3 }
6806 sp-purple ::= OBJECT IDENTIFIER { id-ocfSecurityProfile 4 }
6807
6808 sp-unspecified-v0 ::= ocfSecurityProfileOID (id-sp-unspecified 0)
6809 sp-baseline-v0 ::= ocfSecurityProfileOID {id-sp-baseline 0}
6810 sp-black-v0 ::= ocfSecurityProfileOID {id-sp-black 0}
6811 sp-blue-v0 ::= ocfSecurityProfileOID {id-sp-blue 0}
6812 sp-purple-v0 ::= ocfSecurityProfileOID {id-sp-purple 0}
6813
6814 ocfSecurityProfileOID ::= UTF8String
6815
6816 -- OCF Security Certificate Policies
6817
6818 ocfCertificatePolicy-v1 ::= { id-ocfCertificatePolicy 2}
6819
6820 -- OCF X.509v3 Extensions
6821
6822 id-ocfX509Extensions OBJECT IDENTIFIER ::= { id-OCF 1 }
6823 id-ocfCompliance OBJECT IDENTIFIER ::= { id-ocfX509Extensions 0 }
6824 id-ocfSecurityClaims OBJECT IDENTIFIER ::= { id-ocfX509Extensions 1 }
6825 id-ocfCPLAttributes OBJECT IDENTIFIER ::= { id-ocfX509Extensions 2 }
6826
6827 ocfVersion ::= SEQUENCE {
6828     major    INTEGER,
6829     minor    INTEGER,
6830     build    INTEGER}
6831
6832 ocfCompliance ::= SEQUENCE {
6833     version        ocfVersion,
6834     securityProfile SEQUENCE SIZE (1..MAX) OF ocfSecurityProfileOID,
6835     deviceName     UTF8String,
6836     deviceManufacturer UTF8String}
6837
6838 claim-secure-boot ::= ocfSecurityClaimsOID { id-ocfSecurityClaims 0 }
```

```
6839 claim-hw-backed-cred-storage ::= ocfSecurityClaimsOID { id-ocfSecurityClaims 1 }
6840
6841 ocfSecurityClaimsOID ::= OBJECT IDENTIFIER
6842
6843 ocfSecurityClaims ::= SEQUENCE SIZE (1..MAX) of ocfSecurityClaimsOID
6844
6845 cpl-at-IANAPen ::= OBJECT IDENTIFIER { id-ocfCPLAttributes 0 }
6846 cpl-at-model ::= OBJECT IDENTIFIER { id-ocfCPLAttributes 1 }
6847 cpl-at-version ::= OBJECT IDENTIFIER { id-ocfCPLAttributes 2 }
6848
6849 ocfCPLAttributes ::= SEQUENCE {
6850     cpl-at-IANAPen UTF8String,
6851     cpl-at-model UTF8String,
6852     cpl-at-version UTF8String}
```


6853
6854
6855
6856

Annex E (informative)

Security considerations specific to Bridged Protocols

6857 The text in this Annex is provided for information only. This Annex has no normative impact. This
6858 information is applicable at the time of initial publication and may become out of date.

6859 E.1 Security Considerations specific to the AllJoyn Protocol

6860 This clause intentionally left empty.

6861 E.2 Security Considerations specific to the Bluetooth LE Protocol

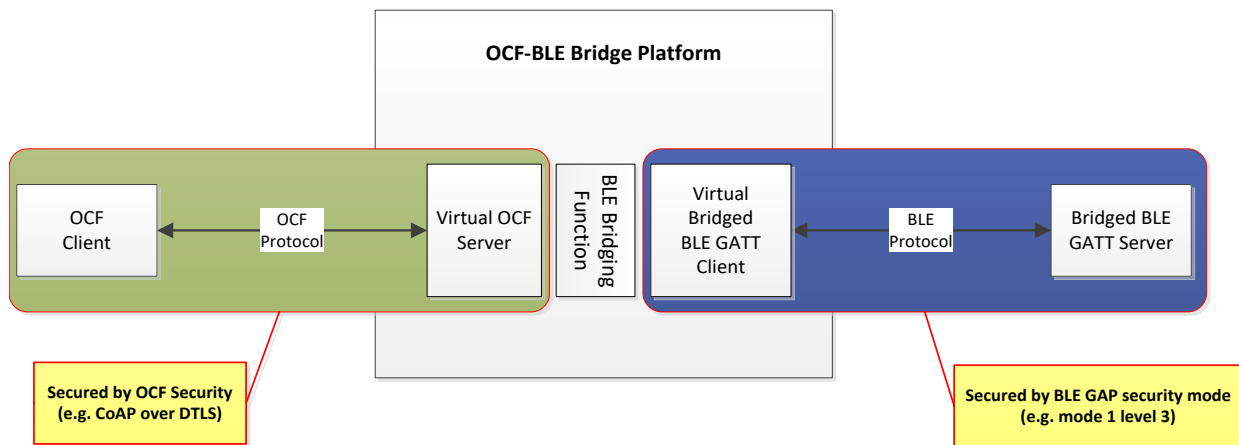
6862 BLE GAP supports two security modes, security mode 1 and security mode 2. Each security mode
6863 has several security levels (see Table E.1)

6864 Security mode 1 and Security level 2 or higher would typically be considered secure from an OCF
6865 perspective. The appropriate selection of security mode and level is left to the vendor.

6866 **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)

6867 Figure E-1 shows how communications in both ecosystems of OCF-BLE Bridge Platform are
6868 secured by their own security.



6869

6870 **Figure E-1 Security Considerations for BLE Bridge**

6871 E.3 Security Considerations specific to the oneM2M Protocol

6872 This clause intentionally left empty.

6873 **E.4 Security Considerations specific to the U+ Protocol**

6874 A U+ server supports one of the TLS 1.2 cipher suites as in Table E.2 defined in IETF RFC 5246.

6875 **Table E.2 TLS 1.2 Cipher Suites used by U+**

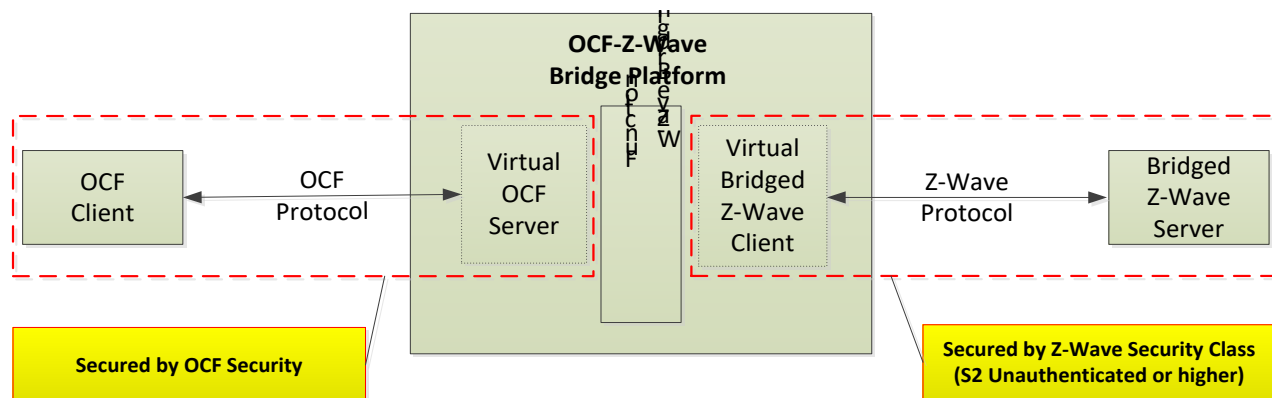
Cipher Suite
TLS_RSA_WITH_AES_128_CBC_SHA256
TLS_RSA_WITH_AES_256_CBC_SHA256
TLS_RSA_WITH_AES_256_CCM
TLS_RSA_WITH_AES_256_CCM_8
TLS_RSA_WITH_AES_256_GCM_SHA384
TLS_DHE_RSA_WITH_AES_256_CBC_SHA256
TLS_DHE_RSA_WITH_AES_256_GCM_SHA384
TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA384
TLS_ECDH_ECDSA_WITH_AES_256_GCM_SHA384
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

6876 The security of the Haier U+ Protocol is proprietary, and further details are presently unavailable.

6877 **E.5 Security Considerations specific to the Z-Wave Protocol**

6878 Z-Wave currently supports two kinds of security class which are S0 Security Class and S2 Security
 6879 Class, as shown in Table E.3. Bridged Z-wave Servers using S2 Security Class for communication
 6880 with a Virtual Bridged Client would typically be considered secure from an OCF perspective. The
 6881 appropriate selection for S2 Security Class and Class Name is left to the vendor.

6882 Figure E-2 presents how OCF Client and Bridged Z-Wave Server communicate based upon their
 6883 own security.



6884

6885

Figure E-2 Security Considerations for Z-Wave Bridge

6886
6887

All 3 types of S2 Security Class such as S2 Access Control, S2 Authenticated and S2 Unauthenticated provides the following advantages from the security perspective;

6888
6889

- The unique device specific key for every secure device enables validation of device identity and prevents man-in-the-middle compromises to security

6890
6891

- The Secure cryptographic key exchange methods during inclusion achieves high level of security between the Virtual Z-Wave Client and the Bridged Z-Wave Server.

6892
6893

- Out of band key exchange for product authentication which is combined with device specific key prevents eavesdropping and man-in-the-middle attack vectors.

6894 See Table E.3 for a summary of Z-Wave Security Classes.

6895

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

6896
6897
6898
6899

On the other hand, S0 Security Class has the vulnerability of security during inclusion by exchanging of temporary 'well-known key' (e.g. 1234). As a result of that, it could lead the disclosure of the network key if the log of key exchange methods is captured, so Z-Wave devices might be no longer secure in that case.

6900 E.6 Security Considerations specific to the Zigbee Protocol

6901
6902
6903

The Zigbee 3.0 stack supports multiple security levels. A security level is supported by both the network (NWK) layer and application support (APS) layer. A security attribute in the Zigbee 3.0 stack, "nwkSecurityLevel", represents the security level of a device.

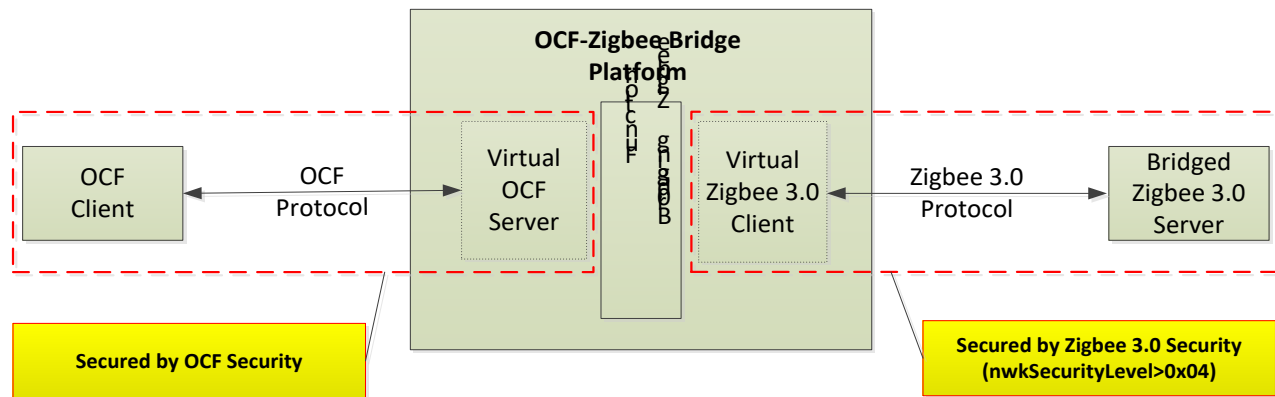
6904 The security level nwkSecurityLevel > 0x04 provides message integrity code (MIC) and/or AES128-
 6905 CCM encryption (ENC). Zigbee Servers using nwkSecurityLevel > 0x04 would typically be
 6906 considered secure from an OCF perspective. The appropriate selection for nwkSecurityLevel is left
 6907 to the vendor.

6908 See Table E.4 for a summary of the Zigbee Security Levels.

6909 **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)

6910 Figure E-3 shows how communications in both ecosystems of OCF-Zigbee Bridge Platform are
 6911 secured by their own security.



6912

6913

Figure E-3 Security Considerations for Zigbee Bridge