



Testing gaps - core components

1 Contents

2	General considerations	2
3	apertis-update-manager	3
4	apparmor	4
5	busybox	4
6	dbus	5
7	gnutls28	6
8	linux-image	6
9	openssl	7
10	ostree	8
11	rust-coreutils	9
12	rust-findutils	9
13	rust-sequoia-sqv	10
14	systemd	11
15	Summary of proposals	11

16 General considerations

17 As described in [Apertis test strategy](#)¹ the approach to do gap analysis is to
18 classify the components under different categories and based on the expected
19 levels of testing for each of them provide a report about the gaps.

20 In general, based on the current workflow, most of the component already meet
21 some standard level of testing, and share some common status which is described
22 below.

23 As a general idea, testing should focus in Apertis specific components and com-
24 ponents with delta from Debian. For components not under heavy development
25 in Apertis the focus should be on integration tests.

26 Please refer to the [Apertis test strategy](#)² for more details on the loops described
27 below.

28 Local loop

29 Developer tests: Required, ad hoc during development.

30 Unit tests: Components with unit test support in Debian inherit this property,
31 each component will be analyzed individually, below.

32 CI loop

33 Linters: Linters are a nice to have feature, but only encouraged for component
34 under development in Apertis.

35 License scan: License scan is already triggered for all the branches in all the
36 packages. This scan is meant to provide a full copyright report available and to

¹<https://www.apertis.org/concepts/distribution/test-strategy/>

²<https://www.apertis.org/concepts/distribution/test-strategy/#loops-and-types>

37 raise a warning in case a license does not match [Apertis license expectations](#)³.

38 OBS build: OBS build is already run in WIP branches/MRs for all the packages.

39 Integration tests: This type of test helps to ensure that proposed changes will not
40 affect the stability of the final solution. Adding this kind of test can make a huge
41 impact by catching regressions earlier, making components under development
42 or with important delta from Debian good candidates. Currently, no package
43 runs integration tests.

44 **Review loop**

45 Review is always required for any kind of change.

46 **Image loop**

47 Installation: Core components are part of daily images and installed on them.

48 License compliance: Core components are already part of daily images and
49 checked as part of the license checks.

50 **Orchestrator loop**

51 Installation: Core components are already part of daily images and installed on
52 them.

53 **Integration tests**

54 Functional tests: This type of test is analyzed on a per component basis.

55 Performance: For the purpose of Apertis as distribution performance is evalu-
56 ated in a high level approach setting time constrains in the functional tests to
57 run, to spot deviation from expected performance behavior.

58 **apertis-update-manager**

Category	Classification
Source	1 - Apertis specific component
Activity	1 - Minimal upstream activity
Commonality	1 - High commonality
Criticality	1 - High criticality
Target	1 - Use in target devices

59 **Local loop**

60 Unit tests: Unit tests not available. A minimal set of test is required to validate
61 that the basic functionality works after changes during development, such as
62 read/write boot count from NVME and UEFI.

³<https://www.apertis.org/policies/license-expectations/>

63 **CI loop**

64 Linters: Linters should be added to ensure all the contributors make changes
65 using the same standards.

66 Integration tests: Since this component is Apertis specific, and has low commu-
67 nity activity, the currently available integration tests are already run as part of
68 CI.

69 **Integration tests**

70 Functional tests: Functional tests are already available, currently 18 test suites
71 run. AUM is one of the most exercised components in all the available architec-
72 tures, including amd64 after adding support to the new reference board.

73 **apparmor**

Category	Classification
Source	2 - Significant delta from Debian
Activity	2 - Medium upstream activity
Commonality	1 - High commonality
Criticality	1 - High criticality
Target	1 - Use in target devices

74 **Local loop**

75 Unit tests: Unit tests already available, currently 71607 tests run. Since this
76 package is not under heavy development improving the coverage is not required.

77 **CI loop**

78 Linters: Linters are not required since this component is not under development
79 inside the Apertis project.

80 Integration tests: Since this component has a significant delta from Debian, the
81 currently available integration tests are already run as part of CI.

82 **Integration tests**

83 Functional tests: Functional tests are already available, currently 14 specific
84 test suites run. The Apparmor is one of the most exercised components in all
85 the available architectures. This component is a special one, since it has to deal
86 with security and is tightly coupled to a desired security configuration. For this
87 reason it is recommended to review periodically the functional tests and adapt
88 them to the needs of downstream distributions and product teams.

89 **busybox**

Category	Classification
Source	2 - Delta from Debian
Activity	2 - Medium upstream activity
Commonality	1 - High commonality
Criticality	1 - High criticality
Target	1 - Use in target devices

90 **Local loop**

91 Unit tests: Unit tests already available, currently 714 tests run. Since this
 92 package is not under heavy development improving the coverage is not required.

93 **CI loop**

94 Linters: Linters are not required since this component is not under development
 95 inside the Apertis project.

96 Integration tests: Recommended given this component has a delta from Debian,
 97 is used during images bootstrap, as well as a partial replacement for **coreutils**,
 98 where changes may affect core functionality. In this context, testing bootstrap-
 99 ping and image creation is a recommended step.

100 **Integration tests**

101 Functional tests: Apertis uses busybox to provide support for some **coreutils**
 102 utilities such as diff, grep and sed, which are used by the standard Debian
 103 utilities to handle packages and also used by tests scripts of different components.
 104 Based on these facts the basic integration tests are already satisfied by the image
 105 generation, for this reason, additional tests are not required.

106 **dbus**

Category	Classification
Source	3 - No delta from Debian
Activity	1 - High upstream activity
Commonality	1 - High commonality
Criticality	1 - High criticality
Target	1 - Use in target devices

107 **Local loop**

108 Unit tests: Unit tests already available, currently around 1986 tests run. Since
 109 this package is not under heavy development, improving the coverage is not
 110 required.

111 **CI loop**

112 Linters: Linters are not required since this component is not under development
113 inside the Apertis project.

114 Integration tests: Not required since this component has high upstream activ-
115 ity/support, there is no delta from Debian and it is not under development.

116 **Integration tests**

117 Functional tests: Apertis already runs the upstream supported **dbus** on daily
118 images. Also, many Apertis components use **dbus** to exchange messages, from
119 this perspective and since this component does not contain any delta from De-
120 bian, adding more integration tests is not required.

121 **gnutls28**

Category	Classification
Source	3 - No delta from Debian
Activity	2 - Medium upstream activity
Commonality	1 - High commonality
Criticality	1 - High criticality
Target	1 - Use in target devices

122 **Local loop**

123 Unit tests: Unit tests already available, currently 746. Since this package is not
124 under heavy development improving the coverage is not required.

125 **CI loop**

126 Linters: Linters are not required since this component is not under development
127 inside the Apertis project.

128 Integration tests: Not required since this component has upstream support,
129 there is no delta from Debian and it is not under development.

130 **Integration tests**

131 Functional tests: Several Apertis components use **gnutls**, as in Debian, however,
132 since it does not contain any delta from Debian, integration tests for itself are
133 not strictly required.

134 **linux-image**

Category	Classification
Source	2 - Latest LTS version
Activity	1 - High upstream activity
Commonality	1 - High commonality

Category	Classification
Criticality	1 - High criticality
Target	1 - Use in target devices

135 **Local loop**

136 Unit tests: Running unit test on linux is very difficult since it is a very low level
 137 package and very dependent on the hardware. Also since this package is not
 138 under development adding unit tests is not required.

139 **CI loop**

140 Linters: Linters are not required since this component is not under development
 141 inside the Apertis project.

142 Integration tests: Linux is a special component as it is very tight to the hardware
 143 and Apertis ships the latest LTS version available for each release. Even if it is
 144 not formally under development in Apertis, configurations are customized and
 145 drivers added to support the reference boards. Given that, the tests to verify
 146 that booting works as expected are already run as part of CI.

147 **Integration tests**

148 Functional tests: Linux is a core component which is tested in every reference
 149 board available to make sure the basic functionality it provides works as ex-
 150 pected. Since different boards use different hardware and drivers, the different
 151 combinations should be tested.

152 These tests are usually tight to high level components which make use of the
 153 functionality. For instance **connman** is the service used to handle network
 154 connections, which makes use of Linux to access the network interfaces. With
 155 this idea in mind integration tests should be divided taking into account tests
 156 for high level components:

- 157 • networking: connman
- 158 • bluetooth: bluez
- 159 • audio: pipewire/gstreamer
- 160 • video: pipewire/gstreamer

161 For basic functionality, like booting, access media devices such as SD card test
 162 are already available.

163 As this component is likely to be customized by downstream distributions, prod-
 164 uct teams should pay special care and adjust tests according to their needs.

165 **openssl**

Category	Classification
Source	2 - Replacement of gnutls
Activity	2 - Medium upstream activity
Commonality	1 - High commonality
Criticality	1 - High criticality
Target	1 - Use in target devices

166 **Local loop**

167 Unit tests: Unit tests already available, currently 2622 tests run. Since this
 168 package is not under development improving the coverage is not required.

169 **CI loop**

170 Linters: Linters are not required since this component is not under development
 171 inside the Apertis project.

172 Integration tests: Several components use **openssl** instead of **gnutls** due to
 173 license restrictions, causing a delta from Debian. For this reason, the currently
 174 available integration tests are already run as part of CI.

175 **Integration tests**

176 Functional tests: Several Apertis components use **openssl** instead of **gnutls** due
 177 to license restrictions, for this reason integration tests for the components with
 178 delta from Debian are recommended. For that, tests based on glib-networking,
 179 which uses openssl, are already available and should cover potential regressions.

180 **ostree**

Category	Classification
Source	2 - Delta from Debian
Activity	2 - Medium upstream activity
Commonality	1 - High commonality
Criticality	1 - High criticality
Target	1 - Use in target devices

181 **Local loop**

182 Unit tests: Unit tests already available, currently 242 which cover the most
 183 common uses.

184 **CI loop**

185 Linters: Linters are not required since this component is not under development
 186 inside the Apertis project.

187 Integration tests: Running integration tests as part of CI is recommended since
188 this package has some delta from Debian and provides critical functionality.

189 **Integration tests**

190 Functional tests: OSTree is tested while testing **Apertis Update Manager**
191 which is the high level application that handles upgrades in Apertis covering
192 the common scenarios, which already has a good testing coverage as previously
193 mentioned.

194 **rust-coreutils**

Category	Classification
Source	2 - Replacement of coreutils
Activity	2 - Medium upstream activity
Commonality	1 - High commonality
Criticality	1 - High criticality
Target	1 - Use in target devices

195 **Local loop**

196 Unit tests: Unit tests are already available, currently 1477 are available but
197 not enabled. The recommendation is to enable them. It is important to note
198 that the support for unit tests in **debcargo** is under development and from the
199 current set of unit tests 1 fails and 27 are ignored.

200 **CI loop**

201 Linters: Linters are not required since this component is not under development
202 inside the Apertis project.

203 Integration tests: Recommended since this package has a delta from Debian, is
204 used during images bootstrap and is a replacement for **coreutils**, where changes
205 may affect core functionality. In this context, testing bootstrapping and image
206 creation is a recommended step.

207 **Integration tests**

208 Functional tests: The basic use of rust-coreutils functionality is indirectly tested
209 by the use of them at different stages, such as image generation and testing
210 scripts. For this reason, additional tests are not required.

211 **rust-findutils**

Category	Classification
Source	2 - Replacement of findutils
Activity	2 - Medium upstream activity

Category	Classification
Commonality	1 - High commonality
Criticality	1 - High criticality
Target	1 - Use in target devices

212 **Local loop**

213 Unit tests: Unit tests are already available, currently 151 tests are available,
 214 but not enabled. The recommendation is to enable them. It is important to
 215 note that the support for unit tests in **debcargo** is under development.

216 **CI loop**

217 Linters: Linters are not required since this component is not under development
 218 inside the Apertis project.

219 Integration tests: Recommended since this package has a delta from Debian, and
 220 it is used as replacement from **findutils**. In this context, testing bootstrapping
 221 and image creation is a recommended step.

222 **Integration tests**

223 Functional tests: The basic use of **rust-findutils** functionality is indirectly
 224 tested by the use of them at different stages, such as image generation and
 225 testing scripts. For this reason, additional tests are not required.

226 **rust-sequoia-sqv**

Category	Classification
Source	2 - Replacement of gpgv
Activity	2 - Medium upstream activity
Commonality	1 - High commonality
Criticality	1 - High criticality
Target	1 - Use in target devices

227 **Local loop**

228 Unit tests: Unit tests are not yet available, however, since this component is
 229 not under development adding them is not required.

230 **CI loop**

231 Linters: Linters are not required since this component is not under development
 232 inside the Apertis project.

233 Integration tests: Recommended since this package has a delta from Debian
 234 and it is used as replacement from **gpgv**. In this context, testing bootstrapping
 235 and image creation is a recommended step.

236 **Integration tests**

237 Functional tests: The basic use of rust-sequoia-sqv functionality is indirectly
238 tested by the use of it at different stages, such as image generation and testing
239 scripts. For this reason, additional tests are not required.

240 **systemd**

Category	Classification
Source	2 - Delta from Debian
Activity	1 - High upstream activity
Commonality	1 - High commonality
Criticality	1 - High criticality
Target	1 - Use in target devices

241 **Local loop**

242 Unit tests: Unit tests already available, currently 596, which cover the most
243 common uses.

244 **CI loop**

245 Linters: Linters are not required since this component is not under development
246 inside the Apertis project.

247 Integration tests: Since this component has some delta from Debian, the current
248 integration tests to verify that booting works as expected are already run as part
249 of CI.

250 **Integration tests**

251 Functional tests: The standard use of **systemd** is tested indirectly by the use
252 of it at different stages, such as booting images and test running services such as
253 **connman** and **pipewire**. Since the delta is more focused in avoiding bashism
254 additional tests are not required.

255 **Summary of proposals**

Component	Test on MR	Bootstrap on MR	Other tests	Comments
apertis-update-manager		Not required	Add unit tests, linters	
apparmor		Not required		Review for downstream
busybox	Not required	Need to add		
dbus	Not required	Not required		
gnutls28	Not required	Not required		
linux-image		Not required		Check high level applic
openssl		Not required		

Component	Test on MR	Bootstrap on MR	Other tests	Comments
rust-coreutils	Not required	Need to add	Enable unit tests	
rust-findutils	Not required	Need to add	Enable unit tests	
rust-sequoia-sqv	Not required	Need to add	Enable unit tests	
systemd		Need to add		
