



Status Page Review

1	Contents	
2	Introduction	2
3	Terminology and concepts	2
4	Use cases	2
5	Non-use cases	2
6	Requirements	2
7	Existing systems	2
8	Self-hosted	3
9	Static	3
10	Dynamic	3
11	Hosted	3
12	Approach	4
13	Evaluation Report	4
14	Monitored services	4
15	Tool comparison	5
16	Recommendation	6
17	Risks	7

18 Introduction

19 As interest and use of Apertis grows it is becoming increasingly important to
20 show the health of the Apertis infrastructure. This enables users to proactively
21 discover the health of the resources provided by Apertis and determine if any
22 issues they may be having are due to Apertis or their infrastructure.

23 Terminology and concepts

- 24 • **Hosted:** Service provided by an external provider that can typically be
25 accessed over the internet.
- 26 • **Self-hosted:** Service installed and run from computing resources directly
27 owned by the user.

28 Use cases

- 29 • A developer is releasing a new version of a package they maintain, but the
30 upload to OBS is failing and they need to find out if it is a misconfiguration

31 on their part or if the OBS service actually down.

32 Non-use cases

- 33 • Providing the Apertis system administrators with a granular over-view of
34 the infrastructure state.

35 Requirements

- 36 • An automated system monitoring status of user accessible resources pro-
37 vided by the Apertis platform.
- 38 • The system displays a simple indication of the availability of the resources.
- 39 • The chosen system appears to be actively maintained:
40 – Hosted services have activity on their website in the last six months
41 – Self-hosted projects show signs of activity in the six months
- 42 • (Optional) The system is hosted on a distinct infrastructure to reduce
43 shared infrastructure that could lead to inaccurate results.

44 Existing systems

45 Numerous externally hosted services and open source projects are available
46 which provide the functionality required to show a status page.

47 Self-hosted

48 The self-hosted options fall into 2 categories:

- 49 • **Static:** The status page is generated to html pages, stored on a web server
50 which then provides the latest status page when requested.
- 51 • **Dynamic:** The page is generated via a web scripting language on the
52 server and served to the user per request.

53 These include the following options:

54 Static

- 55 • [Statusfy](#)¹
- 56 • [ClearStatus](#)²
- 57 • [CState](#)³
- 58 • [status.sh](#)⁴
- 59 • [uptime](#)⁵

¹<https://marquez.co/statusfy>

²<https://github.com/weeblrpress/clearstatus/>

³<https://github.com/cstate/cstate>

⁴https://github.com/Cyclenerd/static_status

⁵<https://uptime.js.org/>

60 **Dynamic**

- 61 • [Cachet](#)⁶
- 62 • [Gatus](#)⁷

63 **Hosted**

64 Many of the hosted services understandably charge a fee to provide a status
65 page. A small number have free options which provide a basic service. As we
66 are looking for a simple option and as a self-hosted option is expected to cost
67 us very little once setup, we will only be considering the free services. The
68 following options have been found:

- 69 • [Better Uptime](#)⁸
- 70 • [Freshstatus](#)⁹
- 71 • [HetrixTools](#)¹⁰
- 72 • [Instatus](#)¹¹
- 73 • [Nixstats](#)¹²
- 74 • [Pagefate](#)¹³
- 75 • [Squadcast](#)¹⁴
- 76 • [StatusKit](#)¹⁵
- 77 • [StatusCake](#)¹⁶
- 78 • [UptimeRobot](#)¹⁷

79 **Approach**

80 As there are an abundance of tools and services available which provide status
81 page functionality, choosing from these existing solutions will be preferred over
82 a home grown solution, assuming that one can be found to fit our requirements,
83 with a home grown solution only considered if none of the existing solutions are
84 appropriate. Our approach is to:

- 85 • Determine services that need to be monitored, this will be critical to dis-
86 count some of the free services that limit the number of services that can
87 be monitored.
- 88 • Each option will be evaluated against the following criteria:

⁶<http://cachethq.io/>

⁷<https://github.com/TwinProduction/gatus>

⁸<https://betteruptime.com/status-page>

⁹<https://www.freshworks.com/status-page/>

¹⁰<https://hetrixtools.com/pricing/uptime-monitor/>

¹¹<https://instatus.com/>

¹²<https://nixstats.com/>

¹³<https://pagefate.com/>

¹⁴<https://www.squadcast.com/>

¹⁵<https://statuskit.com/>

¹⁶<https://www.statuscake.com/features/uptime/>

¹⁷<https://uptimerobot.com/status-page/>

- 89 – Tool provides automated update to status of monitored services
- 90 – Tool can be used to monitor all services that we wish to monitor
- 91 (preferably with some capacity to monitor more in the future if de-
- 92 sired).
- 93 – Simple interface, providing clear picture of status.
- 94 – The tool is actively maintained, either appearing to have active con-
- 95 tributions or in the case of services activity on its website.

96 Evaluation Report

97 Monitored services

98 The following services could be monitored to gauge the status of the Apertis
99 project:

- 100 • **GitLab**: This is the main service used by Apertis developers which hosts
101 the source code used and developed as part of the project.
- 102 • **Website**: This is the main site at www.apertis.org¹⁸. This is hosted by
103 GitLab pages which is a distinct from the main GitLab service.
- 104 • **APT repositories**: This service hosts the `.deb` packages that are build
105 by the Apertis project. This is required in order to build images or up-
106 date/extend existing apt based installations.
- 107 • **Artifacts hosting**: This is where the images built by Apertis are stored
108 along with the OSTree repositories. This service is therefore important
109 for anyone wanting to install a fresh copy of Apertis or update one based
110 on OSTree.
- 111 • **OBS**: Apertis utilizes Collabora’s instance of the Open Build Service.
112 This performs compilation of the source into `.deb` packages. Whilst this
113 will not be directly interacted with by most users, it is required to be
114 available for updates to be generated when releases are made to packages
115 in GitLab and there may be some cases where advanced users may need
116 access to OBS.
- 117 • **LAVA**: Apertis utilizes Collabora’s instance of LAVA. This is primarily
118 used to test images built by Apertis and is thus a critical part of the
119 automated QA infrastructure.
- 120 • **lavaphabridge**: This records the outcome of LAVA runs and displays
121 the test cases used for QA.
- 122 • **hawkBit**: This is a deployment management system that is being inte-
123 grated into Apertis. It provides both a web UI and rest API. Both of
124 these should be monitored.

125 Whilst this list could arguably be reduced a little to just target core services,
126 it would be prudent to choose a service that would allow Apertis room to grow
127 and add services that need monitoring.

¹⁸<http://www.apertis.org>

128 **Tool comparison**

129 The following table was created whilst evaluating the options listed under ex-
 130 isting systems. To save time, where it was apparent that the option was not
 131 going to meet the initial criteria, no further attempt was made to evaluate later
 132 criterion, hence the lack of answers on less suitable options.

Tool	Hosting	Automated	8+ Services?	Simplicity	Activity
UptimeRobot ¹⁹	Service	Yes	Yes - 50	Simple	Active
status.sh ²⁰	Self	Yes	Yes - Unlimited	Simple	Active
Gatus ²¹	Self	Yes	Yes - Unlimited	Simple	Active
Better Uptime ²²	Service	Yes	Yes - 10	Moderate	Active
uptime ²³	Self	Yes	Yes - Unlimited	Moderate	Active
HetrixTools ²⁴	Service	Yes	Yes - 15	Complex	?
StatusCake ²⁵	Service	Yes	Yes - 10	?	Active
Pagefate ²⁶	Service	?	?	-	-
Nixstats ²⁷	Service	?	No - 5	-	-
Statusfy ²⁸	Self	No	Yes - Unlimited	-	-
ClearStatus ²⁹	Self	No	Yes - Unlimited	-	-
CState ³⁰	Self	No	Yes - Unlimited	-	-
Cachet ³¹	Self	No	yes - Unlimited	-	-
Freshstatus ³²	Service	No - Requires freshping	-	-	-
Instatus ³³	Service	No - Requires extra service	-	-	-
Squadcast ³⁴	Service	No	?	-	-
StatusKit ³⁵	Service	No	?	-	-

133 **Recommendation**

134 Based on the above evaluation, the top 4 options would appear to be:

¹⁹<https://uptimerobot.com/status-page/>
²⁰https://github.com/Cyclenerd/static_status
²¹<https://github.com/TwinProduction/gatus>
²²<https://betteruptime.com/status-page>
²³<https://uptime.js.org/>
²⁴<https://hetrixtools.com/uptime-monitor/>
²⁵<https://www.statuscake.com/features/uptime/>
²⁶<https://pagefate.com/>
²⁷<https://nixstats.com/>
²⁸<https://marquez.co/statusfy>
²⁹<https://github.com/weeblrpress/clearstatus/>
³⁰<https://github.com/cstate/cstate>
³¹<http://cachethq.io/>
³²<https://www.freshworks.com/status-page/>
³³<https://instatus.com/>
³⁴<https://www.squadcast.com/>
³⁵<https://statuskit.com/>

- 135 • Better Uptime
- 136 • Gatus
- 137 • status.sh
- 138 • UptimeRobot

139 The choice can be further slimmed by making a decision between a service and
140 a self-hosted solution.

141 A self-hosted solution has the advantage that it will remain available long-term,
142 not being reliant on an outside provider, however they will also require main-
143 tenance and up keep. An externally provided service has the advantage that
144 it is hosted on distinct infrastructure from that hosting the other Apertis ser-
145 vices and thus less likely to be made unavailable by a fault affecting the whole
146 platform. An external service is also likely to provide a more independent and
147 reliable evaluation of the platform status.

148 Based on this our recommendation would be to utilize UptimeRobot to provide
149 a status page for Apertis.

150 **Risks**

- 151 • UptimeRobot stops providing free service: In the event that the free ser-
152 vice ceases to be offered or changes such that it is no longer suitable to
153 Apertis, it would appear to be fairly trivial to migrate to an alternative
154 service or decide to self-host.