

V10-SP1 服务器版 COCKPIT 部署手册

内部资料 请勿外传

文档属性

客户名称		文件类别	技术文档
文件名	V10-SP1 服务器版 COCKPIT 部署手册	是否保密	否
编制者	邹建林	编制者职位	售后工程师
编制者邮箱	zoujianlin@kylinos.cn	编制日期	2021-08-19
版本修订记录			
版本号	修订时间	修订说明	
V1.0	2021-08-19	新建	

一、 文档说明.....	4
二、 背景.....	4
三、 方案.....	6

内部参考 请勿外传

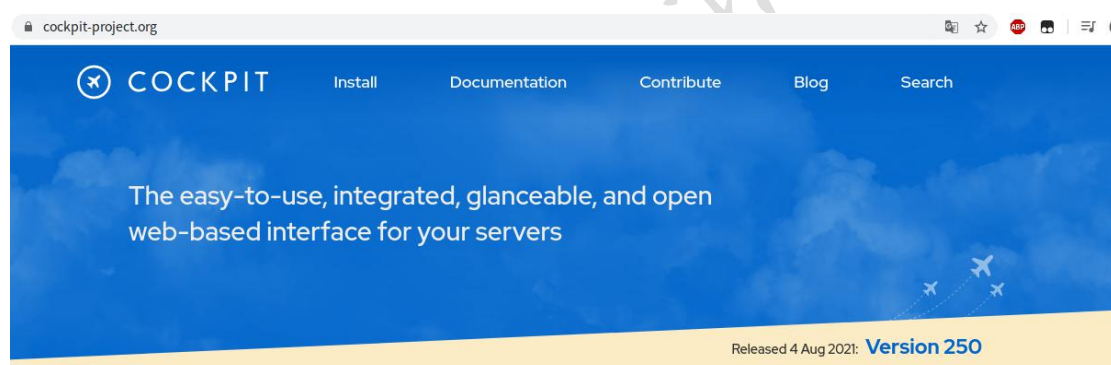
一、文档说明

本文档基于系统版本为 Kylin-Server-10-SP1-Release-Build04-20200711-x86_64，所使用 cockpit 版本为 cockpit-196.3-1.ky10.ky10.x86_64。

二、背景

cockpit 是一个免费且开源的基于 web 的管理工具，系统管理员可以执行诸如存储管理、网络配置、检查日志、管理容器等任务。通过 cockpit 提供的友好的 Web 前端界面可以轻松地管理我们的 GNU/Linux 服务器，非常轻量级，Web 界面也非常简单易用。更重要的是通过 cockpit 可以实现集中式管理。

项目地址：<https://cockpit-project.org/>



cockpit 具有如下功能：

- Inspect and change network settings
- Configure a firewall
- Manage storage (including RAID and LUKS partitions)
- Create and manage virtual machines
- Download and run containers
- Browse and search system logs
- Inspect a system's hardware
- Upgrade software
- Keep tabs on performance
- Manage user accounts
- Inspect and interact with systemd-based services
- Use a terminal on a remote server in your local web browser
- Switch between [multiple Cockpit servers](#)
- Extend Cockpit's functionality by installing [a growing list of apps and add-ons](#)
- [Write your own custom modules](#) to make Cockpit do anything you want

查看和配置网络、防火墙配置

存储管理、创建和管理虚拟机

下载和运行容器、浏览和搜索系统日志

查看系统硬件、更新软件

系统性能监控、用户账户管理

基于 **systemd** 管理服务、远程管理其他服务器

功能可扩展

并可以帮助问题诊断：

诊断网络问题

监控异常虚拟机并做出反应

监控 SELinux 日志并修复常见违规行为

监控 CPU 负载、内存使用情况、网络活动和存储性能等

V10-SP1-Server-0711-x86_64 系统已经预装 **cockpit** 及其基础组件：

```
[kylin@localhost 桌面]$ rpm -qa | grep cockpit
cockpit-system-196.3-1.ky10.ky10.noarch
cockpit-storaged-197.3-1.ky10.p01.ky10.noarch
cockpit-bridge-196.3-1.ky10.ky10.x86_64
cockpit-196.3-1.ky10.ky10.x86_64
cockpit-ws-196.3-1.ky10.ky10.x86_64
cockpit-packagekit-197.3-1.ky10.p01.ky10.noarch
[kylin@localhost 桌面]$
```

并支持如下扩展，通过 **yum** 安装即可：

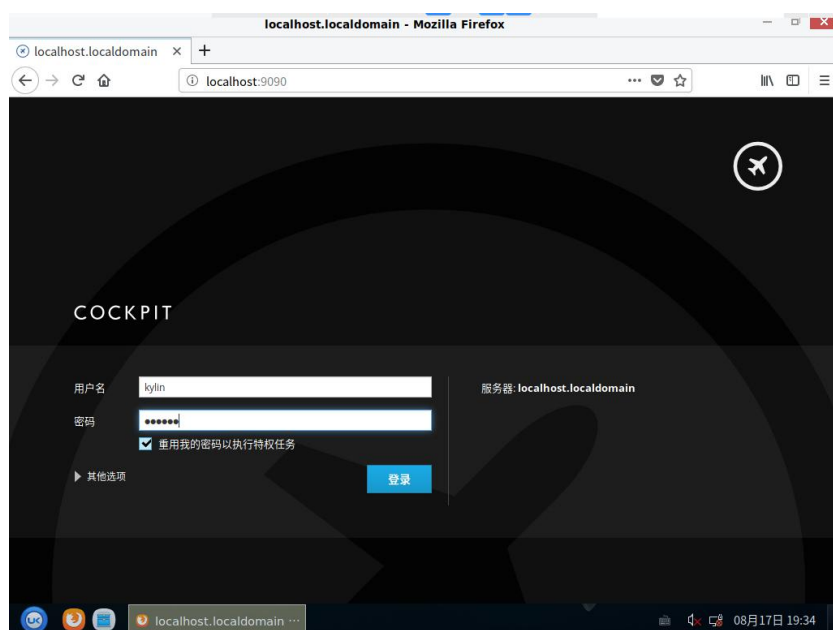
```
[kylin@localhost 桌面]$ yum search cockpit-*
Kylin Linux Advanced Server 10 - 0s      58 kB/s | 10 MB      02:59
Last metadata expiration check: 0:01:43 ago on 2021年08月17日 星期二 14时56分12秒.
===== Name Matched: cockpit-* =====
cockpit-ws.x86_64 : Cockpit Web Service
cockpit-doc.noarch : Cockpit deployment and developer guide
cockpit-pcp.x86_64 : Cockpit PCP integration
cockpit-tests.x86_64 : Tests for Cockpit
cockpit-bridge.x86_64 : Cockpit bridge server-side component
cockpit-system.noarch : Cockpit admin interface package for configuring and
                        : troubleshooting a system
cockpit-storaged.noarch : Cockpit user interface for storage, using udisks
cockpit-machines.noarch : Cockpit user interface for virtual machines
cockpit-dashboard.noarch : Cockpit remote servers and dashboard
cockpit-packagekit.noarch : Cockpit user interface for packages
[kylin@localhost 桌面]$
```

三、方案

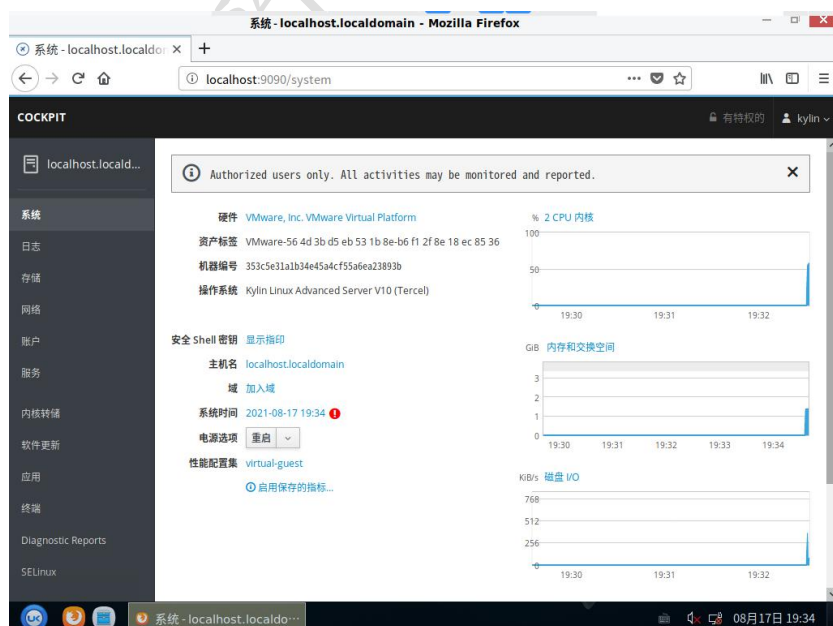
3.1 简单使用

cockpit 的使用较为便捷友好，图形可视化操作，此处仅做简单介绍。

默认端口为 9090，浏览器访问 <http://localhost:9090/> 或 <http://127.0.0.1:9090/> 即可访问本机 cockpit。

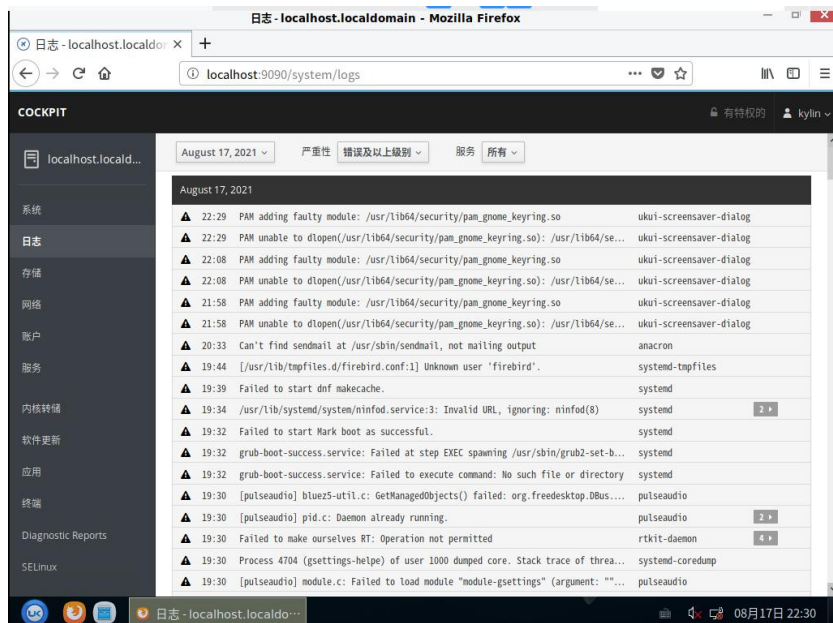


使用系统账号密码登录即可进入 cockpit 页面（系统管理员账户建议勾选下方“重用我的密码以执行特权任务”，作用相当于 `sudo`），默认展示系统概况

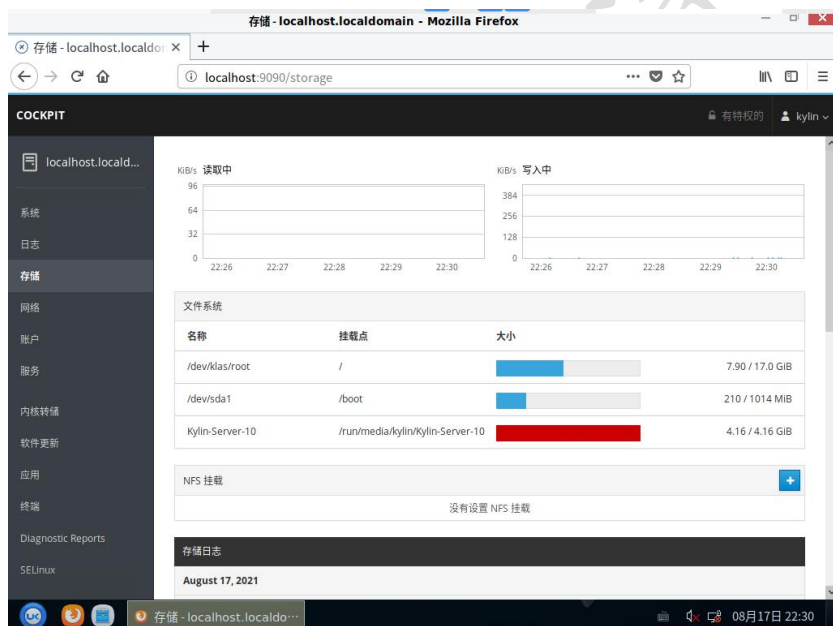


点击左侧菜单栏选项即可进入相应栏目进行浏览和操作，例如：

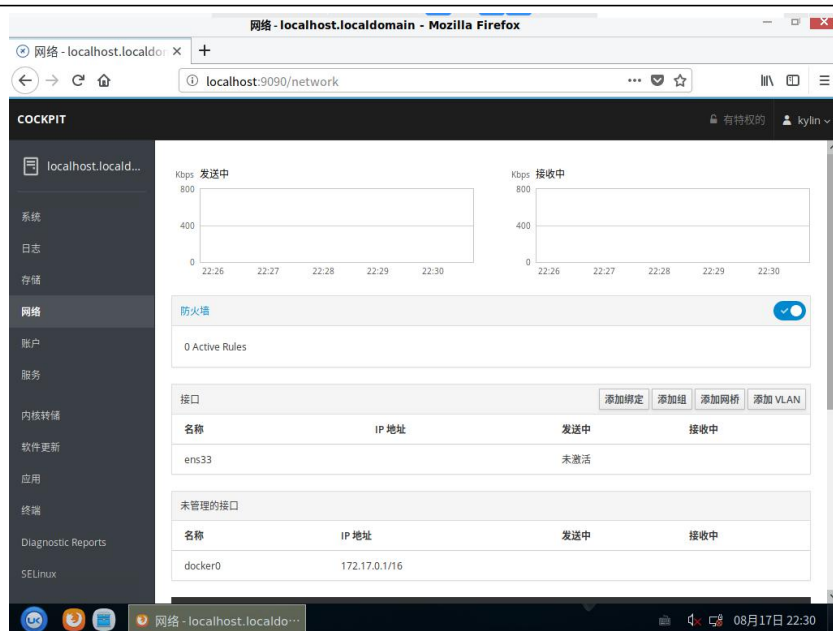
日志：



存储：



网络：



3.2 二次开发

cockpit 的页面可以作为 frame 嵌套在开发者自定义的网页中，并且可以按需修改已有功能组件，也可以自行扩展新增功能模块。

开发者文档：

<https://cockpit-project.org/guide/latest/development>

cockpit 单独页面的链接组成：

`http://主机: 端口/cockpit/@localhost/包名/组件.html` 文件

All resource URLs are under the `/cockpit` namespace. In cases where a Cockpit component is being embedded the `/cockpit` may be followed by a plus sign and another `embedder` specific identifier.

What follows is either a `@host` or `$checksum` which tells cockpit where to find the package. Checksums are used when more than one host has identical packages and the resources can be cached.

The `package` name is next, followed by the `component` HTML path inside that package. And lastly a hash allows for navigation within a single component. The hash should follow a URL path and/or query string form.

示例：

<https://localhost:9090/cockpit/@localhost/system/logs.html>

对应：

预装组件存储位置为 `/usr/share/cockpit`，所以上述链接对应

/usr/share/cockpit/systemd/下 logs 相关文件 logs.html.gz, logs.min.js.gz (及 manifest.json 中的声明)

```
[kylin@localhost systemd]$ pwd
/usr/share/cockpit/systemd
[kylin@localhost systemd]$ ls
hwinfo.css.gz      po.cs.js.gz      po.ko.js.gz      services.css.gz
hwinfo.html.gz     po.de.js.gz      po.pl.js.gz      services.html.gz
hwinfo.min.js.gz   po.es.js.gz      po.pt_BR.js.gz   services.min.js.gz
index.html.gz      po.fi.js.gz      po.ru.js.gz      system.css.gz
logs.html.gz       po.fr.js.gz      po.sv.js.gz      system.min.js.gz
logs.min.js.gz     po.it.js.gz      po.uk.js.gz      terminal.css.gz
manifest.json      po.ja.js.gz      po.zh_CN.js.gz   terminal.html.gz
po.ca.js.gz        po.js.gz         po.zh_TW.js.gz   terminal.min.js.gz
```

主要页面链接:

cockpit 主页:

<http://localhost:9090>

运行概况 (CPU、内存、磁盘、网络):

<http://localhost:9090/cockpit/@localhost/system/index.html>

系统信息:

<http://localhost:9090/cockpit/@localhost/system/hwinfo.html>

日志:

<http://localhost:9090/cockpit/@localhost/system/logs.html>

终端:

<https://localhost:9090/cockpit/@localhost/system/terminal.html>

服务:

<http://localhost:9090/cockpit/@localhost/system/services.html>

存储:

<http://localhost:9090/cockpit/@localhost/storage/index.html>

账户:

`http://localhost:9090/cockpit/@localhost/users/index.html`

链接可以直接访问，也可以作为 `iframe` 插入所需页面：

```
<iframe width="800px" height="600px"
src="https://localhost:9090/cockpit/@localhost/system/logs.html"/>
```

cockpit 的包识别路径是 `~/.local/share/cockpit`、`/usr/local/share/cockpit` 和

`/usr/share/cockpit`。cockpit 安装后预装组件存放位置是 `/usr/share/cockpit`，如需修改预装组件则修改该目录下对应组件的文件即可（主要是 `.html`，`.css`，`.js`）。

```
[kylin@localhost cockpit]$ pwd
/usr/share/cockpit
[kylin@localhost cockpit]$ ls
apps          kdump          packagekit     shell          static         tuned
basel         motd           realmd         sosreport     storaged       users
branding      networkmanager selinux        ssh           systemd
```

cockpit 也可以按照规范自己新建组件，下面以 `pinger` 包为例进行开发：

新建工作目录：

```
mkdir -p ~/.local/share/cockpit/pinger
```

写 `manifest.json` 声明文件文件

`manifest.json` 说明：<https://cockpit-project.org/guide/latest/packages.html#package-manifest>

```
kylin@Kylin:~/.local/share/cockpit/pinger$ cat manifest.json
{
  "version": 0,
  "tools": {
    "pinger": {
      "label": "Pinger",
      "path": "pinger.html"
    }
  }
}
```

写页面文件，并引入 `cockpit.css` 和 `cockpit.js`

```
kylin@Kylin:~/local/share/cockpit/pinger$ cat pinger.html
<!DOCTYPE html>
<html>
<head>
  <title>Pinger</title>
  <meta charset="utf-8">
  <link href="../base1/cockpit.css" type="text/css" rel="stylesheet">
  <script src="../base1/cockpit.js"></script>
</head>
<body>
  <div class="container-fluid">
    <table class="form-table-ct">
      <tr>
        <td><label class="control-label" for="address">Address</label></td>
        <td><input class="form-control" id="address" value="8.8.8.8"></td>
      </tr>
      <tr>
        <td><button class="pf-c-button pf-m-primary" id="ping">Ping</button></td>
        <td><span id="result"></span></td>
      </tr>
    </table>
    <pre id="output"></pre>
  </div>
  <script src="pinger.js"></script>
</body>
</html>
```

写页面的js文件:

```
kylin@Kylin:~/local/share/cockpit/pinger$ cat pinger.js
const address = document.getElementById("address");
const output = document.getElementById("output");
const result = document.getElementById("result");
const button = document.getElementById("ping");

function ping_run() {
  cockpit.spawn(["ping", "-c", "4", address.value])
    .stream(ping_output)
    .then(ping_success)
    .catch(ping_fail);

  result.innerHTML = "";
  output.innerHTML = "";
}

function ping_success() {
  result.style.color = "green";
  result.innerHTML = "success";
}

function ping_fail() {
  result.style.color = "red";
  result.innerHTML = "fail";
}

function ping_output(data) {
  output.append(document.createTextNode(data));
}

// Connect the button to starting the "ping" process
button.addEventListener("click", ping_run);

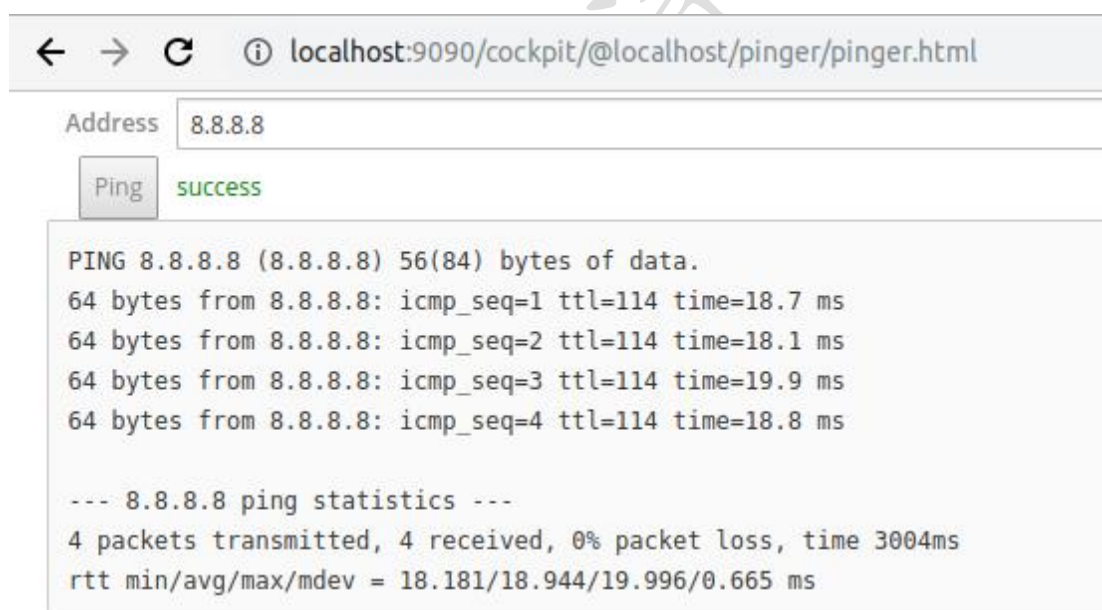
// Send a 'init' message. This tells integration tests that we are ready to go
cockpit.transport.wait(function() { });
```

运行 cockpit-bridge --packages 即可看到新增的包


```
kylin@kylin:~/local/share/cockpit/pinger$ cockpit-bridge --packages
apps: /usr/share/cockpit/apps
base1: /usr/share/cockpit/base1
dashboard: /usr/share/cockpit/dashboard
docker: /usr/share/cockpit/docker
domain: /usr/share/cockpit/realmd
machines: /usr/share/cockpit/machines
network: /usr/share/cockpit/networkmanager
pcp: /usr/share/cockpit/pcp
performance: /usr/share/cockpit/tuned
pinger: /home/kylin/.local/share/cockpit/pinger
playground: /usr/share/cockpit/playground
shell: /usr/share/cockpit/shell
ssh: /usr/share/cockpit/ssh
storage: /usr/share/cockpit/storaged
system: /usr/share/cockpit/systemd
updates: /usr/share/cockpit/packagekit
users: /usr/share/cockpit/users
zoner: /home/kylin/.local/share/cockpit/zoner
```

访问按照规则组建的链接即可使用：

<http://localhost:9090/cockpit/@localhost/pinger/pinger.html>



在 cockpit 网页端刷新之后即可看到菜单栏已经新增 Pinger

localhost.locald...

存储

网络

账户

服务

内核转储

软件更新

时区

应用

终端

Dagnostic Reports

Pinger

SELinux

Address

8.8.8.8

Ping

success

PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=128 time=23.4 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=128 time=21.4 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=128 time=51.0 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=128 time=27.5 ms

--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 21.391/30.811/50.951/11.833 ms