1. 10-second history of cloud computing
2. Infrastructure aaS: my other computer is a data center
Your turn-key Cockpit UI in a CI/CD ecosystem

1. Platform aaS: Kubernetes
Your turn-key Cockpit UI in a CI/CD ecosystem

1. Software aaS: we don’t host our source repos any more, GitHub
Your turn-key Cockpit UI in a CI/CD ecosystem

1. undeniably the pinnacle of evolution: Cockpit Continuous Integration and Deployment aaS
2. that’s what I introduce today
Your turn-key Cockpit UI in a CI/CD ecosystem

1. Conceptually: Linux session running in a web browser; technically very similar to ssh/VT/GNOME login
2. Aimed at admins who are new to Linux, e.g. coming from the Windows world and familiar with the concepts, but not Linux terminology
3. but also to experienced ones for infrequent tasks (set up RAID once a year, don’t remember all the commands); not just setup, but also investigating “what is wrong with this machine”
4. apt or yum install away in Fedora, Atomic, RHEL, Debian, Ubuntu, Arch
1. System page: Summary information about the machine and its current status
2. can drill down into more detailed graphs and information.
3. Menu on the left shows available administration pages for this machine, and can switch between multiple machines
1. subpage of Networking is a UI for firewalld
1. See and interact with your local libvirt or ovirt VMs
2. Cockpit team maintains pages seen on the screenshots
1. there will always be things missing for your use cases, so designed from the ground up to be easily extensible
2. offers JS API to interact with connected target machine: programs, D-Bus, files, sockets, etc.
Your turn-key Cockpit UI in a CI/CD ecosystem

1. little example: create a UI for ping
2. input for address, button to start, and pre for output
1. wire cockpit API for running a process - ping in this case to this UI
2. whenever something new on stdout → append to output field for live streaming
3. slightly simplified, e.g. no error handling, but this is the gist
4. similar structure for a D-Bus call, or files
Your turn-key Cockpit UI in a CI/CD ecosystem

1. initially looks like this; enter address, press button
and you see the result
2. appears in the menu via a little declaration file called manifest; not shown here
3. above good enough for your own personal environment/company specific pages
4. ex: cheap monitoring/control of services or house automation
5. cockpit more popular, more extension projects which are public, get packaged and team-maintained
6. ex: UI for podman, building installable OS images, IPA server, Fleet Commander
7. proposed: NFS server, SSL certificate management
8. then tossing the above into a single HTML file is not good enough
Public projects

1. Separation of HTML, CSS, and JavaScript into lots of little files for maintainability
2. Don’t do UI by hand like in pinger, integrate React and PatternFly
3. JavaScript toolchain to compile all your files into a blob the browser can understand
4. complex build system, integrate static code checks
5. create automated browser tests, run them in PRs
6. test on various operating systems, maintain VM images for these
7. release very often to GitHub, various distros, COPR, dockerhub, update your project page, etc.
8. putting this together is a daunting task
Your turn-key Cockpit UI in a CI/CD ecosystem

1. we put together the Cockpit starter kit, does all that for you
2. best practices for a Cockpit project
3. example UI with all the glory I mentioned before
4. devel-install: run straight out of your build tree; install: /usr/local/, build rpm
1. looks unspectacular, but demonstrates cockpit API (reading hostname) and LESS/CSS
2. point is to be a simple React component which you can directly hack on without worrying about all the boilerplate
Your turn-key Cockpit UI in a CI/CD ecosystem

Integration testing

1. RPM build, integration test
2. test looks simple, but does a lot of stuff for you
3. download appropriate Cockpit VM image (lots of OSes), builds code, installs it into the VM, starts headless Chromium, runs your test on it
4. re-uses VMs of Cockpit team, half-time job to maintain them
5. integrate into CI: webhook, ask Cockpit team to whitelist to run on our infra
Automated releases

1. release process: push a signed git tag with a summary of changes
2. our cockpituous infra then builds release tarballs, srpms, pushes them to github, Fedora, dockerhub, copr, etc.
3. real file has lots of comments
4. just like with CI, ask Cockpit team
5. that part is relatively easy to self-host: container with a bunch of credentials; or run on your laptop
1. routine maintenance tasks: latest NPM dependencies, uploading translation templates to Zanata, download translations
2. bots for code maintenance; example for NPM update
3. proposes a PR for updating to latest React, tests pass; human can sign off and presses the button
Your turn-key Cockpit UI in a CI/CD ecosystem

Current users

1. these projects are real-life, thus this is not a pipe dream; let's add your's
2. Our team wants to scale from "we build UIs for everything" to "we support your team with building your UI"
3. we work a lot on providing CI infrastructure, cross-project testing and maintenance
Your turn-key Cockpit UI in a CI/CD ecosystem

1. Home page leads to mailing lists, documentation
2. Join us on the hackfest on Sunday
3. thanks for your attention; Q+A