

Red Hat Enterprise Linux 8 の Web Console (Cockpit)

森若 和雄

Solution Architect

2019-09

このスライドの位置付けと目的

- 対象
 - GUIでLinuxを管理できると嬉しい人
 - Linuxのデスクトップ環境を持ってないのでXが必要なGUIツールを使うのは難しい人
- 目的
 - RHELに同梱されていて各種サーバ管理に利用できるWeb UIを提供するCockpitを紹介

概要

- Web Console (Cockpit)とは?
- Cockpit Packageによる機能拡張
- Cockpitのセキュリティ

Web Console



サーバ管理用のWeb UIを提供

- サービス管理
- アカウント管理
- ネットワーク管理
- ファイアウォール管理
- ストレージ管理
- ログ閲覧
- 仮想マシン管理
- 仮想端末 などの管理機能を提供

<https://cockpit-project.org/>

Copyright Red Hat K.K. All rights reserved.

The screenshot displays the Cockpit web console interface for a Red Hat Enterprise Linux system. The left sidebar contains navigation links for System, Logs, Storage, Networking, Accounts, Services, Diagnostic Reports, SELinux, Applications, Kernel Dump, Subscriptions, Software Updates, and Terminal. The main content area shows two line graphs for '読み取り中' (Read) and '書き込み' (Write) I/O, both with a scale from 0 to 96 KiB/s. Below the graphs is a 'ファイルシステム' (Filesystem) table:

名前	マウントポイント	Size
/dev/rhel/root	/	4.75 / 7.99 GiB
/dev/vda1	/boot	165 / 1014 MiB

Below the table is an 'NFS マウント' (NFS Mount) section with a message: 'NFS マウントが設定されていません' (NFS mount is not configured). The 'ストレージログ' (Storage Log) section shows a log for November 28, 2018, with entries for 'udisks' and 'libudisks2'. The 'RAID デバイス' (RAID Devices) section shows a message: 'RAID として設定されたストレージがありません' (No storage configured as RAID). The 'ボリュームグループ' (Volume Groups) section shows a table with one entry: 'rhel' with a size of 9.00 GiB. The 'VDO デバイス' (VDO Devices) section shows a message: 'VDO として設定されたストレージがありません' (No storage configured as VDO). The 'iSCSI ターゲット' (iSCSI Targets) section shows a message: 'iSCSI ターゲットが設定されていません' (No iSCSI targets configured). The 'ドライブ' (Drives) section shows a table with two entries: 'VirtIO Disk' (10 GiB Hard Disk) and 'QEMU DVD-ROM (QM00001)' (Optical Drive).

Cockpitの特長

- 独自のDBや設定ファイルなどをほぼ持たない
 - dbus、ファイルアクセス、コマンド実行などへのアクセスを提供するバックエンドとブラウザからアクセスするjavascript APIを基盤とする
 - アクセス制御はPolicyKitにより実施
- 利用していない時にリソースを事実上消費しない
- プラグイン形式での機能追加が容易
 - ~/.local/ 以下に配置することで特定ユーザのみ利用可能
- Kerberosを利用したSSOに対応

Cockpitのインストール

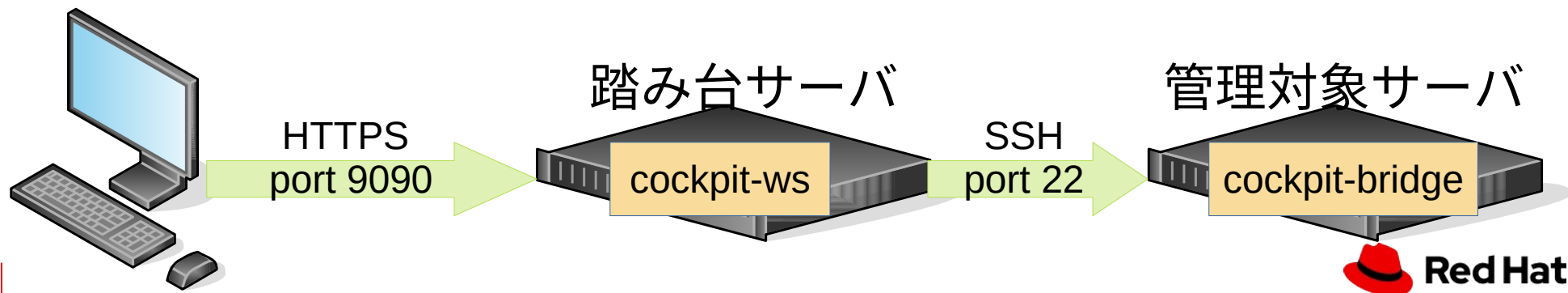
- パッケージ導入
 - `yum install -y cockpit cockpit-dashboard`
- 証明書設置
 - `cp cockpit.cert /etc/cockpit/ws-certs.d/`
- ファイアウォール設定
 - `firewall-cmd --add-port=9090/tcp`
 - `firewall-cmd --permanent --add-port=9090/tcp`
- 起動設定
 - `systemctl enable cockpit.socket`
 - `systemctl start cockpit.socket`
- ブラウザで接続
 - `firefox https://rhel8.corp.example.com:9090/`

TLS用にCA局発行の
証明書を配置します。
なければ自己署名
証明書が自動生成
されます。

cockpitそのものの
設定をしなくても
動作します

リモート管理

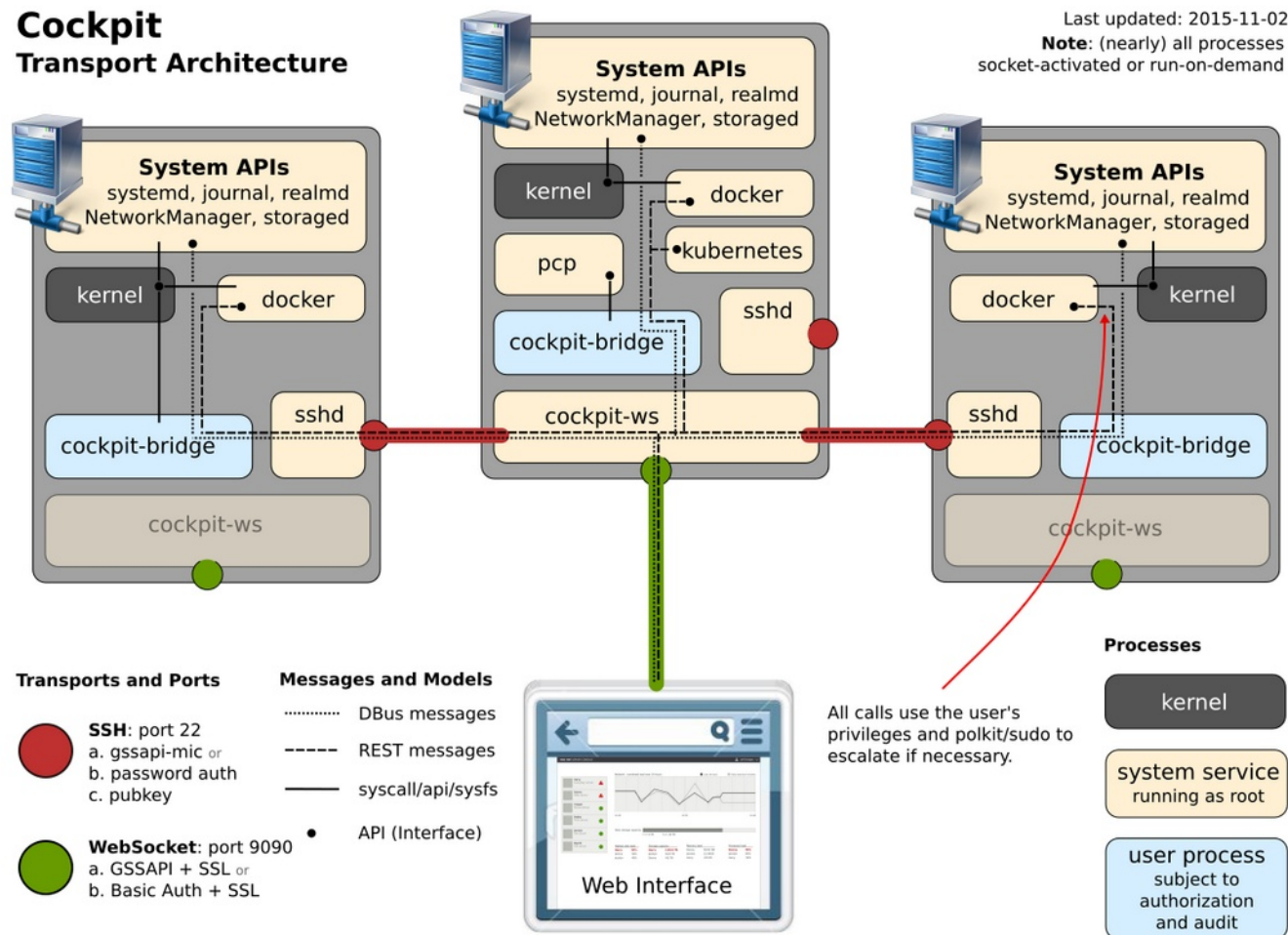
- cockpitはブラウザが直接接続できないシステムも管理できる (cockpit-dashboardパッケージが必要)
- cockpit同士でsshによる通信をおこなう
 - 踏み台サーバからsshによる接続ができればリモートのcockpitと直接通信できなくても管理できる
 - cockpitへログインしたユーザの権限でssh接続



Cockpit Packageによる機能拡張

Cockpitの 基本動作

- cockpit-wsが待受け
- ブラウザでログイン
- cockpit-bridgeがユーザ権限で起動
- cockpit-bridgeがdbus, pcp, プロセス実行などを行う
- リモートへはsshで接続



Cockpitを拡張するには？

- プラグイン(Cockpit Package)による拡張に対応
- dbusでのリクエストやコマンド実行、ファイル読み出し等を行うjavascript用APIを提供

コマンド実行

```
cockpit.spawn(["ping", "8.8.8.8"])
```

ホスト名取得

```
proxy = cockpit.dbus("org.freedesktop.hostname1").proxy()
```

ファイル内容取得

```
file = cockpit.file("/etc/motd", {})
```

Cockpit Packageとは?

以下のようなファイル群を配置することでCockpit内で利用する画面を作成する。

`/usr/share/cockpit/`

`hoge/`

パッケージ用ディレクトリ

`manifest.json`

メニュー内の場所、
ラベルや関連ファイルなどを指定

`hoge.html`

表示するhtml

`hoge.js`

操作を実装するjavascript

Cockpit Packageの配置場所

- ファイルを配置する場所が3箇所あり、オーバーライドできる
 1. /usr/share/cockpit/* (rpm用、システム全体用)
 2. /usr/local/share/cockpit/* (システム全体用)
 3. ~/.local/share/cockpit/* (特定ユーザ用)
- cockpit-bridge --packages コマンドで(実行したユーザにとっての) Cockpit Package一覧を表示
- manifest.jsonで同じ名前を宣言した場合、manifest.json内のpriorityの値が大きい方が優先される

個人用にCockpit Packageを作ってみる

- インストール先ディレクトリ作成
 - `mkdir -p ~/.local/share/cockpit/`
- cockpitのソースコードを取得
 - `git clone https://github.com/cockpit-project/cockpit.git`
- シンボリックリンク作成
 - `cd cockpit/examples/`
 - `ln -s ${PWD}/pinger ~/.local/share/cockpit/`
- ブラウザでcockpitへ接続(または再ログイン)
 - メニュー上の“Pinger”メニューでpingコマンド実行



snake.usersys.r...



サービス

389 Directory Server



Diagnostic Reports

Pinger

SELinux

Starter Kit

アプリケーション

カーネルダンプ

ソフトウェア更新

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=119 time=4.91 ms
```

```
64 bytes from 8.8.8.8: icmp_seq=2 ttl=119 time=4.97 ms
```

```
64 bytes from 8.8.8.8: icmp_seq=3 ttl=119 time=5.87 ms
```

```
64 bytes from 8.8.8.8: icmp_seq=4 ttl=119 time=4.87 ms
```

```
--- 8.8.8.8 ping statistics ---
```

```
4 packets transmitted, 4 received, 0% packet loss, time 3003ms
```

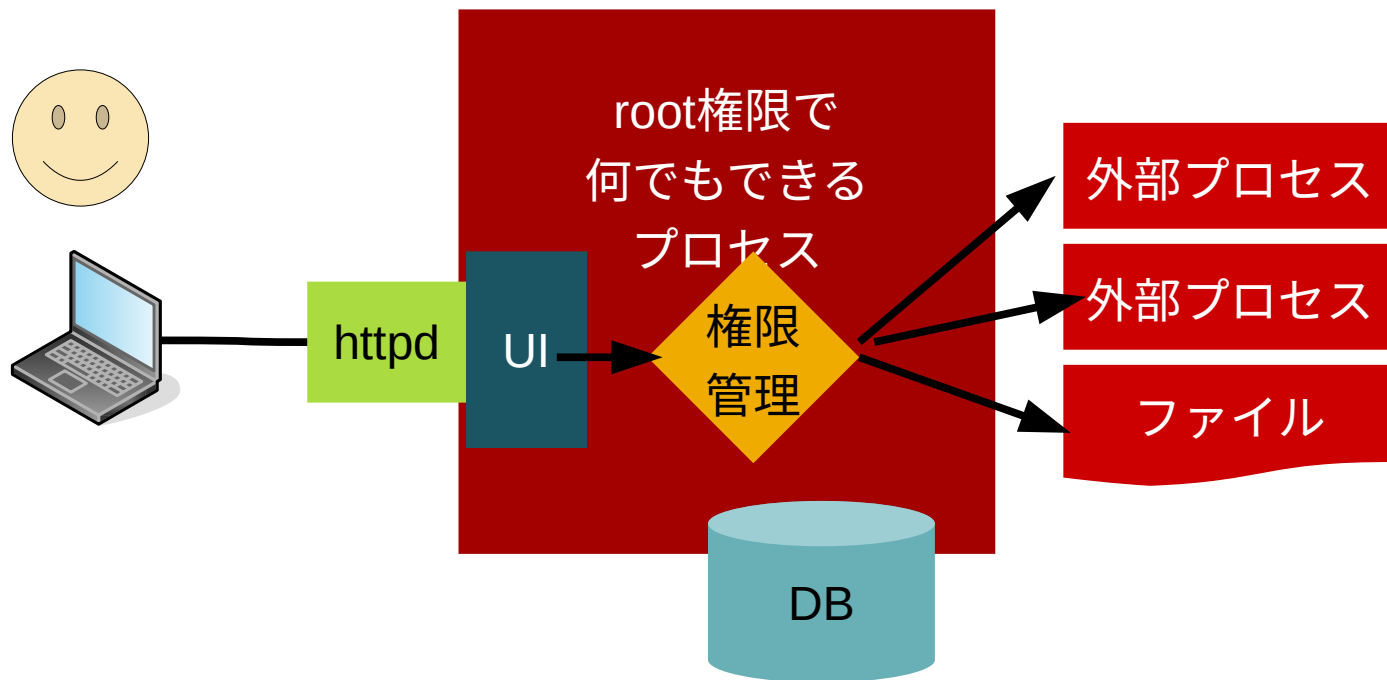
```
rtt min/avg/max/mdev = 4.874/5.162/5.879/0.418 ms
```

Cockpit Packageの 参考文献

- cockpit projectのTutorialカテゴリ
 - <https://cockpit-project.org/blog/category/tutorial.html>
- cockpitプロジェクトのソースコード
 - <https://github.com/cockpit-project/cockpit/>
- 本格的なpluginのひな型 “Starter Kit”
 - rpmパッケージ化、テストケース実行, 一般公開に対応
 - <https://github.com/cockpit-project/starter-kit>
- Cockpitドキュメント内 “Developer Guide”
 - <https://cockpit-project.org/guide/latest/development.html>

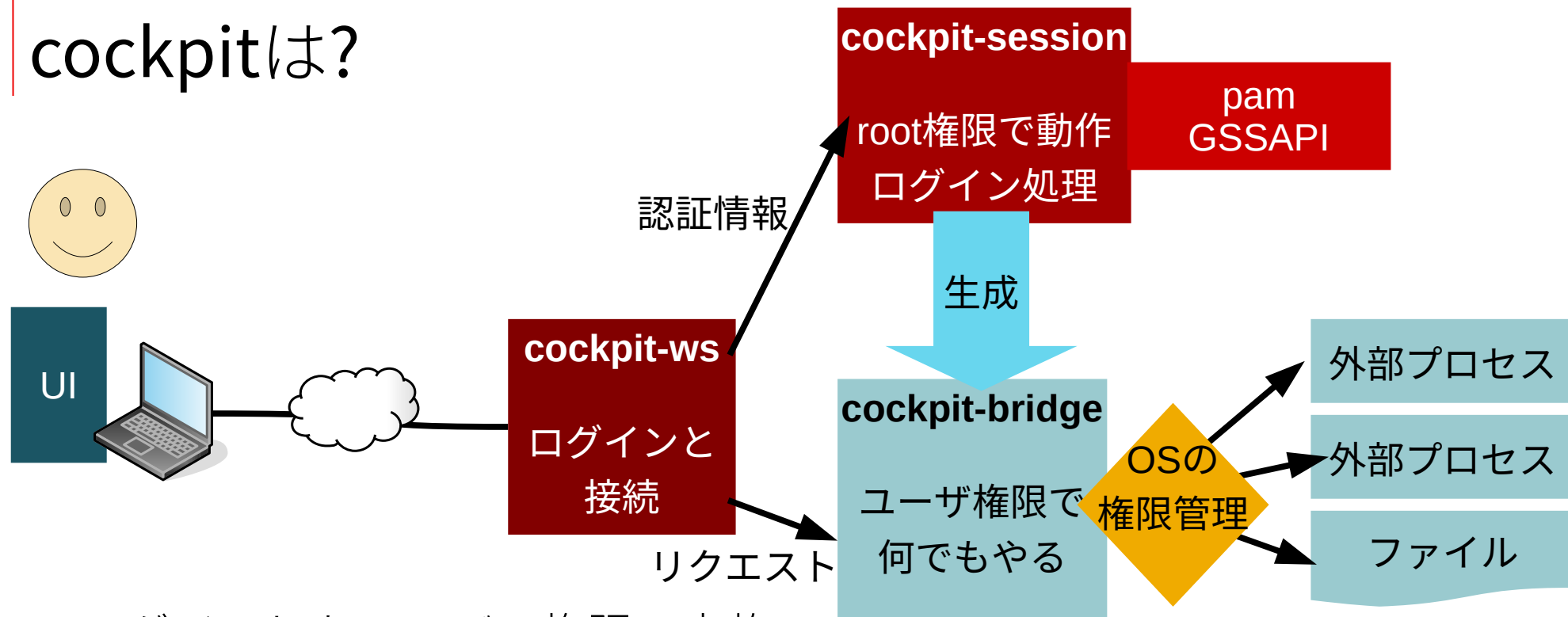
Cockpitのセキュリティ

よくある(?) 「Webコンソール」のイメージ(偏見)



- 外部コマンド実行やファイル変更をroot権限で実施
- 「root権限で何でもできるプロセス」
 - 複雑になり、バグが深刻な問題につながる
- 独自の権限管理設定やDBが必要

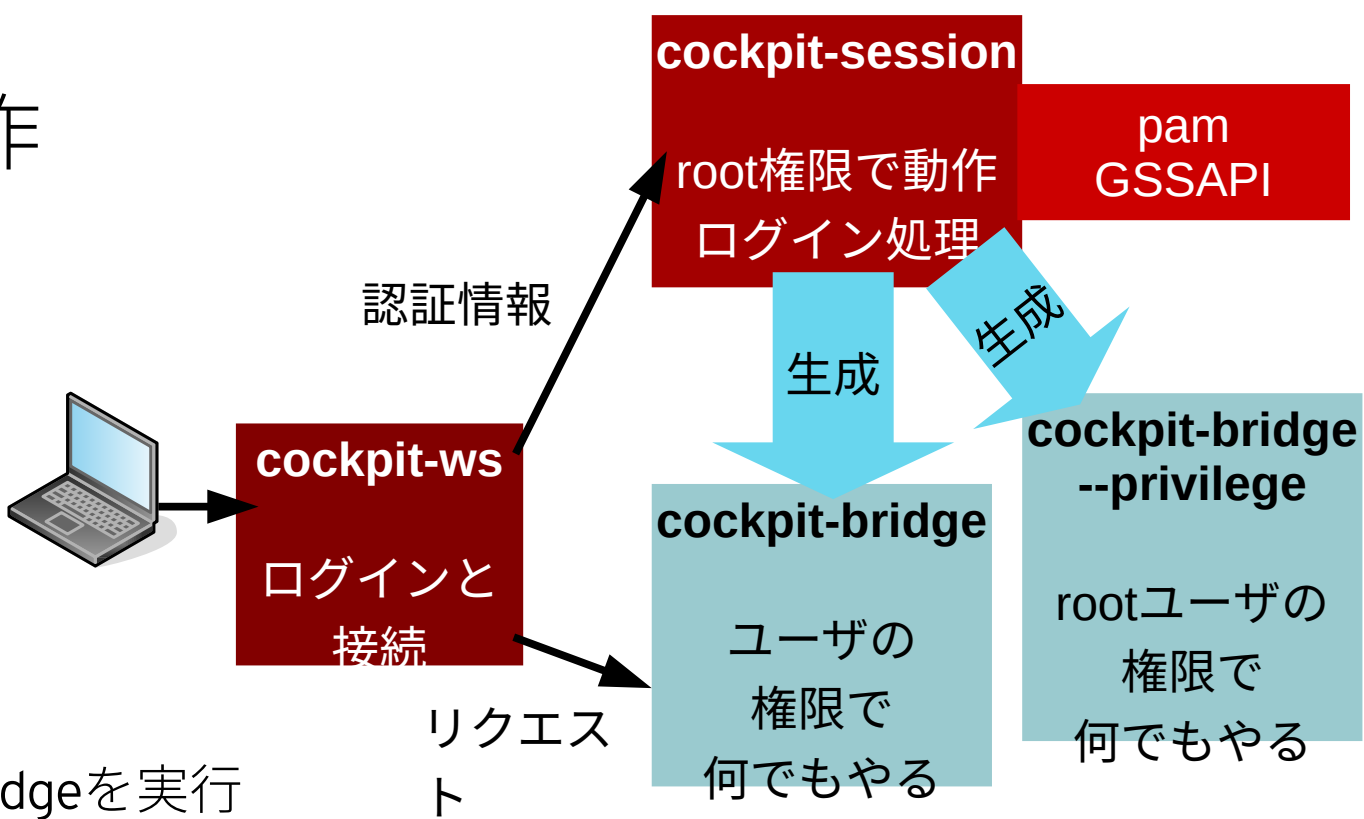
cockpitは?



- ログインしたユーザの権限で実施
- root権限利用はセッション開始時のみ
- OSの権限管理をそのまま使う

ログイン時の動作

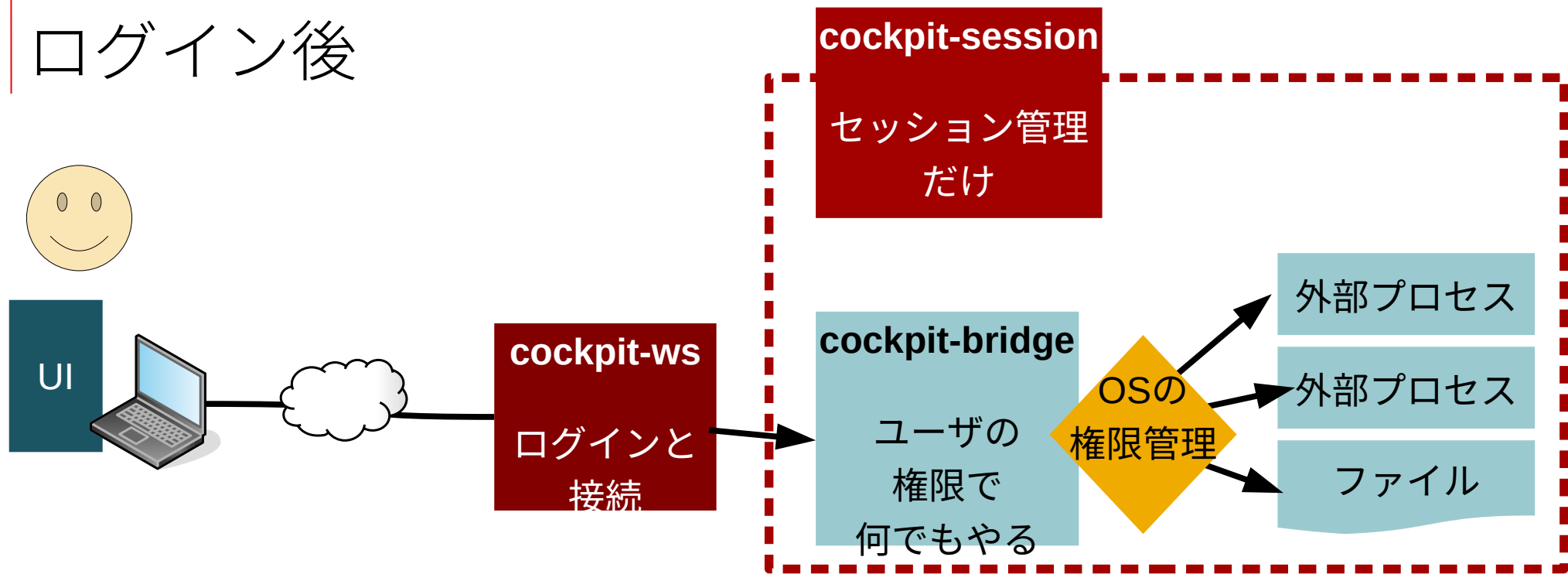
1. cockpit-wsがログイン画面を表示
2. 認証情報を送付
 - ID+Password
 - GSSAPI
3. cockpit-sessionを実行
4. PAMで認証
5. 認証ユーザでcockpit-bridgeを実行
6. 以下チェックのある場合pkexecでroot権限のcockpit-bridgeを起動



☒ 特権タスクにパスワードを再使用する

デフォルトではwheelグループのユーザのみ成功

ログイン後



- ブラウザ上のjavascriptとcockpit-bridge(およびcockpit-bridge --privilege)が通信
- cockpit-bridge本体と、packageと呼ばれるプラグインがdbus, REST API, プロセス実行など各種の操作を実施。通常のユーザ権限管理をそのまま利用。

Cockpitに関連したセキュリティ上の注意点

- SELinuxのポリシーも準備されているのでできれば使う
 - enforcingで使うことでcockpit-wsやcockpit-sessionに未知の脆弱性があった場合にも任意プロセス実行などを予防する
- 自己署名証明書の利用はお勧めしません
 - ブラウザとcockpit-ws間の通信路が信頼できてはじめて安全な仕組み
 - 後述のIdentity Managementとの統合で証明書の自動発行が可能
- 特権取得にPolicyKitを利用している
 - systemdでも利用するがサーバ管理時に気にしないケースが多いので注意
 - 一部操作を制限したい場合は /etc/polkit-1/rules.d/*.rules でポリシー定義をおこなう。ポリシー制限の粒度はサービスによる。

Identity Managementとの統合

- RHEL同梱のRed Hat Identity Managementと統合されています
- IdMが管理するドメインにサーバが参加していれば以下が可能
 - Web Consoleのリモート管理で利用するssh接続をシングルサインオン(SSO)で処理
 - Web Consoleで利用するサーバ証明書を自動発行し自己署名証明書の利用なし

まとめ

- RHELの管理を簡単にするWeb Consoleは、簡単に利用できるだけでなくセキュリティにも配慮して開発されています
 - ログイン時に最小の特権だけを利用
 - 特権を利用する範囲についてSELinuxで制限が可能
 - 実績があるPolicyKitなど枯れた技術を活用
- Identity Managementと統合されています

Cockpitのセキュリティ 参考文献

- RHEL8ドキュメント「Web コンソールを使用したシステムの管理」
 - <https://bit.ly/rhel8-webconsole-ja2>
- Cockpitプロジェクトblog「Is Cockpit Secure?」
 - <https://cockpit-project.org/blog/is-cockpit-secure.html>
- Cockpitプロジェクトドキュメント「Single Sign On」
 - <https://cockpit-project.org/guide/latest/sso.html>

Thank You