Armadillo Base OSセミナー

株式会社アットマークテクノ

www.atmark-techno.com

目次

第１部：Armadillo Base OSについて

第１部の内容

- IoT機器を運用する為に考える事
- Armadillo Base OSの概要
- コンテナについて

IoT機器向けのOSとは？

IoT機器にはDebian/Ubuntu由来の汎用ディストリビューションが多い

IoT機器から見た汎用ディストリビューション

- デスクトップ用途やサーバ用途に便利に作られている
  ⇒IoT機器としては不要なものが多い（脆弱性の懸念）
- OSのサイズが大きすぎる
  ⇒安さが求められるIoT機器はストレージ容量が限られる
- アップデート機能やリカバリ機能が無い
  ⇒IoT機器が脆弱性に対応していく為には必須
- OSのサポート期間が決まっている(5年程度)
  ⇒IoT機器は5年では運用期間が短すぎる

長期に渡って運用するIoT機器向けのOSには向いていない？

脆弱性対策は？

脆弱性対策を怠ると・・・・

- 脆弱性を狙った攻撃を受ける
- DDoS攻撃を仕掛ける端末として利用される
- 情報を抜き取られる/改竄される

攻撃から守る為には…
脆弱性への対応が必要
（＝アップデート）

常に最新の状態が望ましいが、アップデートには課題が多い

- リモートアップデートの機能が無く手作業
- アップデートが失敗すると現地での作業が必要
- 容易に立ち入ることが出来ない場所にある
- アップデートファイルが大きすぎる
  (通信断、アップデートに時間がかかる)

つまり、
・リモート/ローカルアップデート
・失敗時のリカバリ
・最小単位のアップデート
が出来る事が望ましい

IoT機器を安定運用する為には？

Atmark Techno

できれば他に任せたい事

自分でやらないといけない事

PoC段階では検討されない事

	実運用に必要な内容	PoCでの実施内容
運用	アップデート・保守	アップデート・保守
運用	キッティングオペレーション	キッティングオペレーション
運用	継続供給	継続供給
運用	セキュリティ対策	セキュリティ対策
運用	通信断・パケットロス対策	通信断・パケットロス対策
ソフトウェア	クラウドとの通信	クラウドとの通信
ソフトウェア	データ処理	データ処理
ソフトウェア	現場との通信	現場との通信
ソフトウェア	OS/ファームウェア・ドライバ	OS/ファームウェア・ドライバ
ハードウェア	製品寿命	製品寿命
ハードウェア	電源ノイズ・瞬断対策	電源ノイズ・瞬断対策
ハードウェア	法規制（CE・技適など）	法規制（CE・技適など）
ハードウェア	温度・湿度	温度・湿度
ハードウェア	振動・衝撃	振動・衝撃
ハードウェア	防滴・防塵・防爆	防滴・防塵・防爆

PoCで実施

IoT機器向けの新OS

Atmark Techno

長期安定運用と**セキュリティ対策**が可能な
IoT機器向けのOSとして

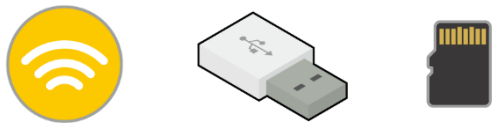
Armadillo Base OS

を新たに開発しました

IoT機器特化の「Armadillo Base OS」

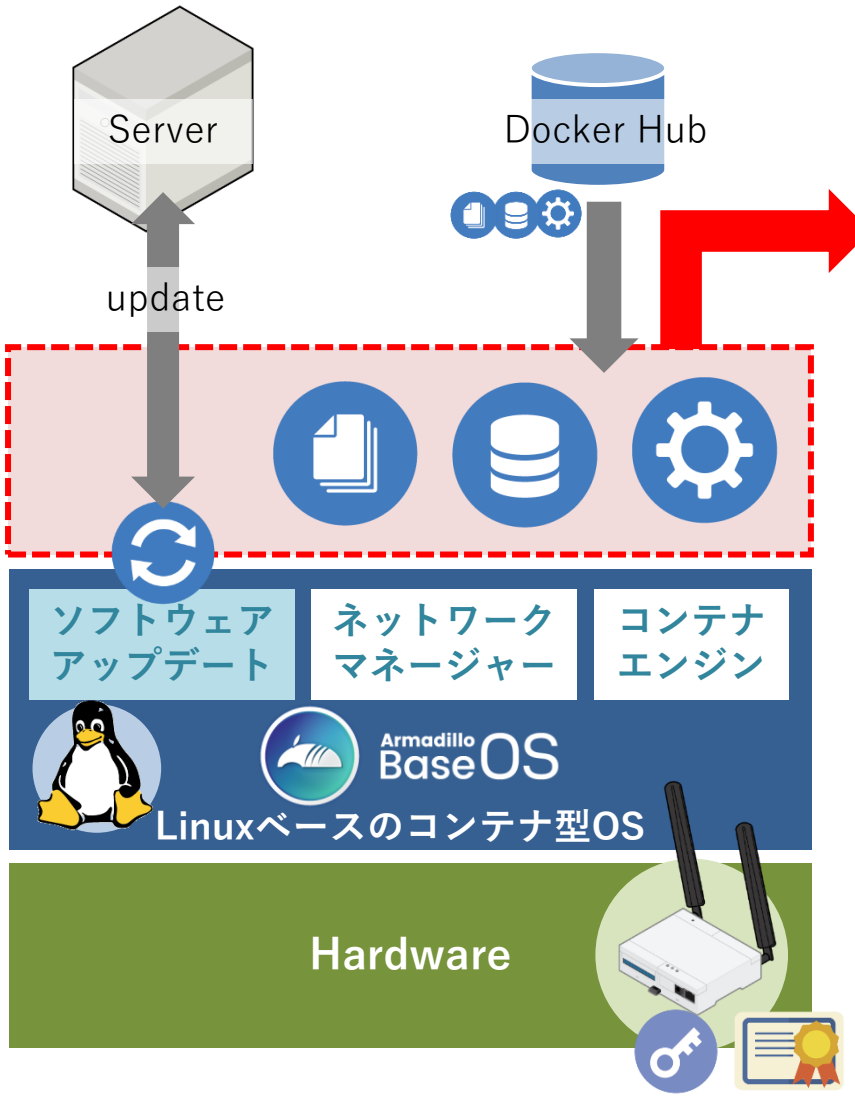
アップデート機能を標準搭載

- イメージ二重化でロールバック可能
  (アップデート失敗でもリカバリー)
- 差分更新機能で最小データ転送
- 多様なアップデート手段
  (ネットワーク/USB/SD)



長期で安定運用するために

- OSの長期メンテナンス
- 突然の電源断でも壊れにくい
  堅牢なファイルシステム
- プログラム領域のリードオンリー化
- 運用ログの記録



モダンなソフトウェア環境

- アプリ開発に集中できる
  (OS部分は最適化済み)
- Docker Hubのコンテナを利用可能
- (A6E)ゲートウェイコンテナを
  使ってIoTアプリも簡単に実現
- (G4/X2)GUIアプリはFlutterで開発

多面的なセキュリティ機能

- OSコンパクト化で脆弱性を限定
- コンテナ構造でサンドボックス化
- セキュアブートで改ざん防止
- セキュアエレメントで個体認証
- OP-TEEのセキュアな実行環境

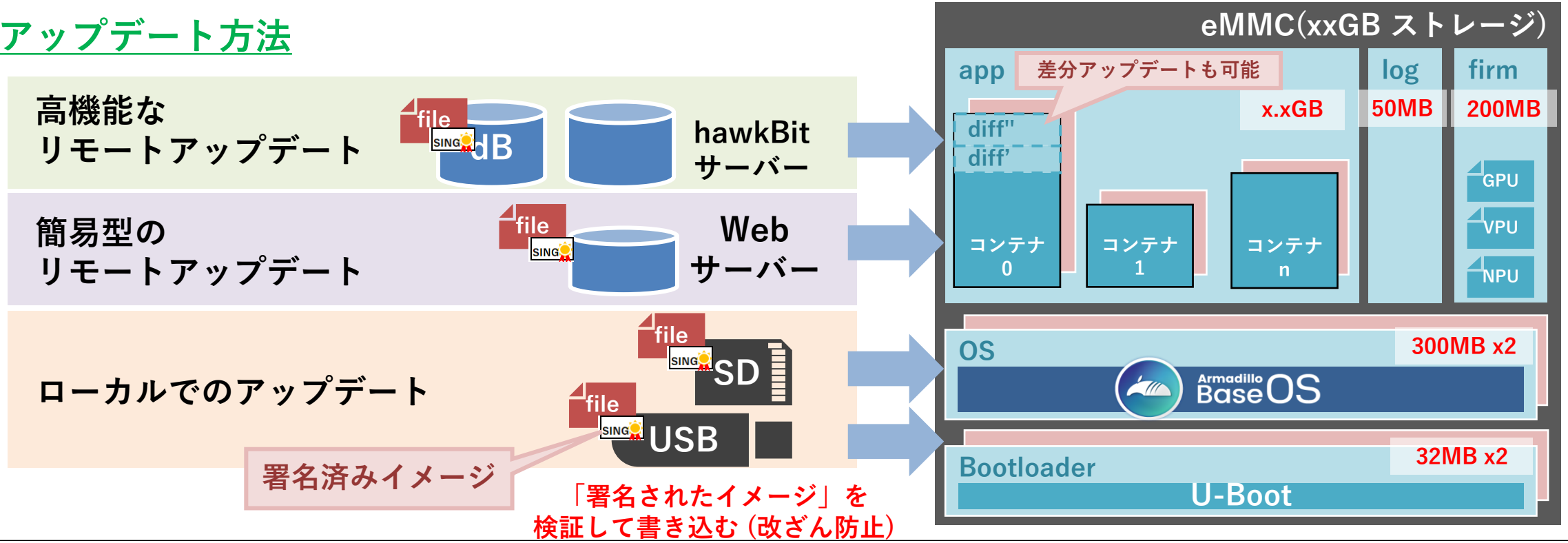
ソフトウェアアップデート機能

ソフトウェアアップデートはリモート/ローカルそれぞれでアップデートが可能
（詳細は第4部で解説）

■アップデート対象

ファイル単体からOSのみ、コンテナ単位やコンテナの差分アップデートも可能

■アップデート方法

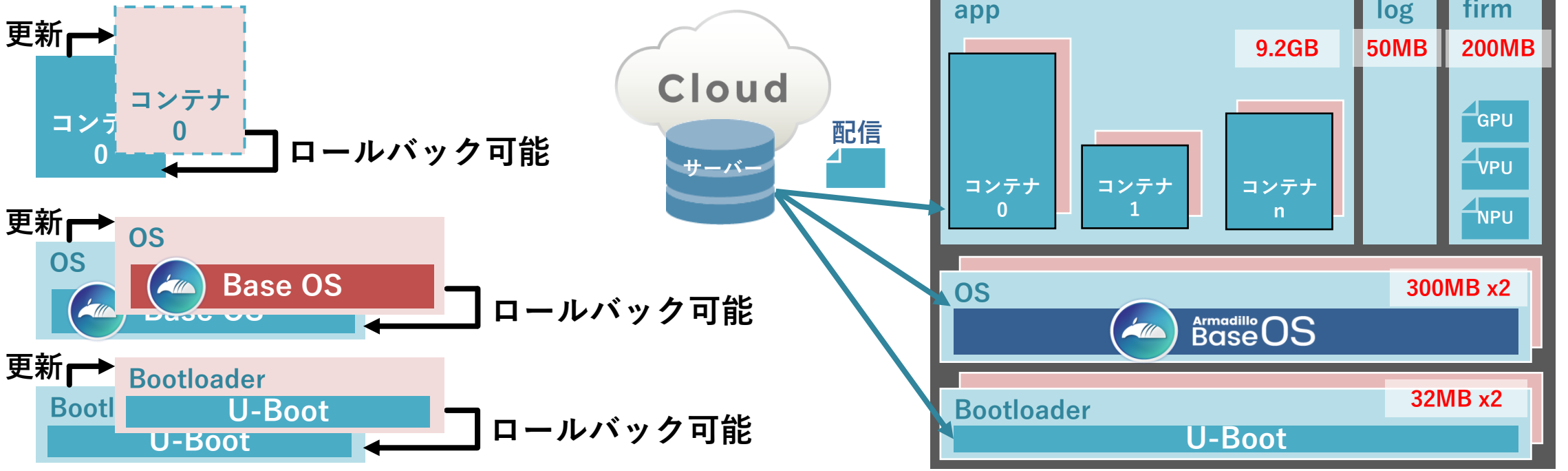


ロールバック機能

ロールバック機能は自動(または手動)でアップデート前のバージョンに戻って起動する仕組み

Armadillo Base OSはコンテナ / OS領域 / Bootloader領域は二面化により1回分のバックアップが存在し、ロールバック機能によりリカバリが可能

アップデート中の電源断などで正常起動できない場合に自動で前のバージョンにロールバック



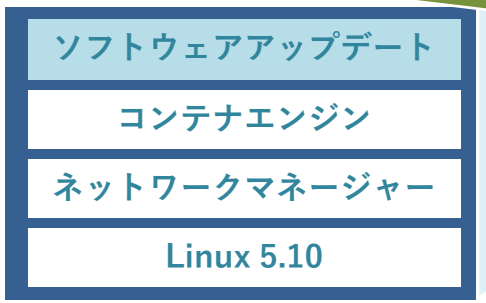
様々なセキュリティ対策

Atmark Techno

アップデート機能の標準化
（脆弱性の回避）

ソフトウェアへの署名
（不正ソフトの書込み禁止）

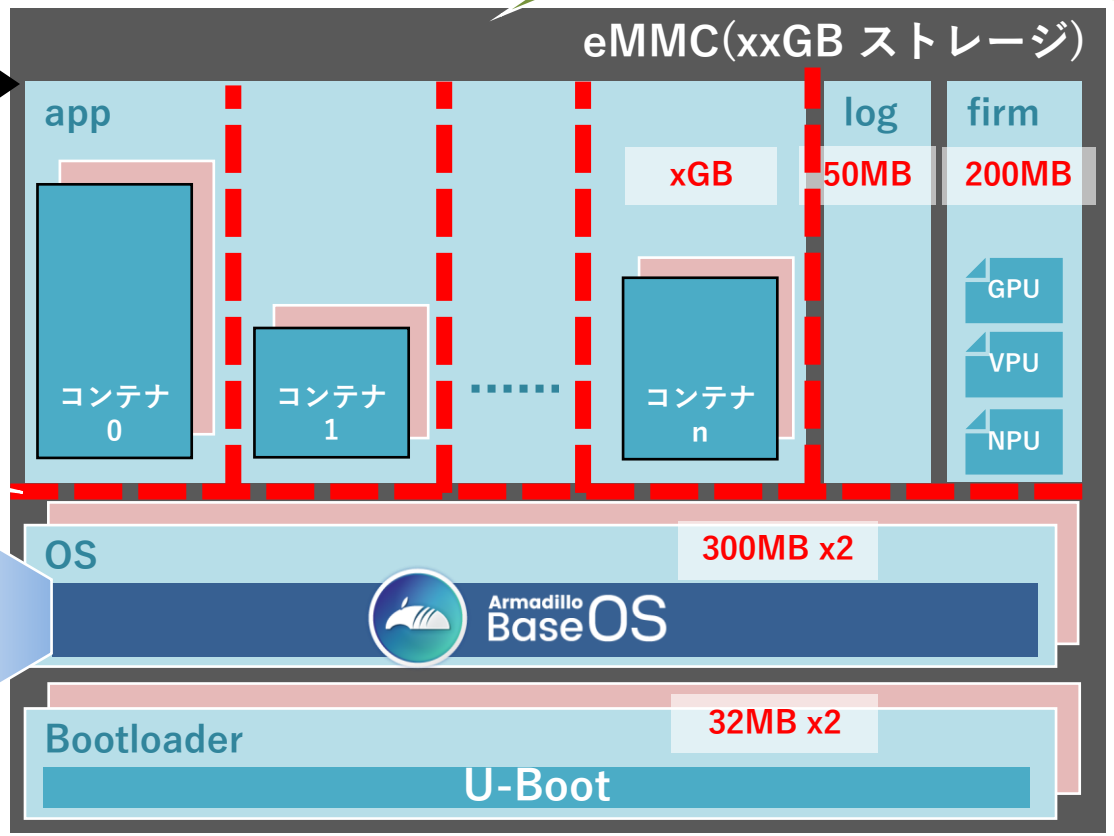
コンテナ構造
（明示的なアクセス権限）



約200MB
（圧縮配信時:約100MB）

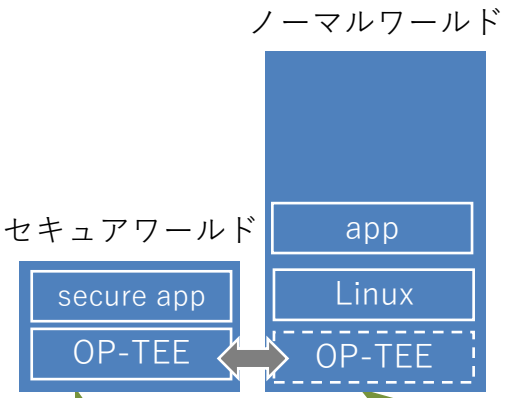
コンパクト化（脆弱性の限定）

コンテナ構造
（サンドボックス化）



セキュアエレメント
（デバイス真正性の証明）

暗号化ストレージ
（プログラム/データ保護）



セキュアブート
（改ざん防止）

OP-TEE
（セキュア空間のアプリ）

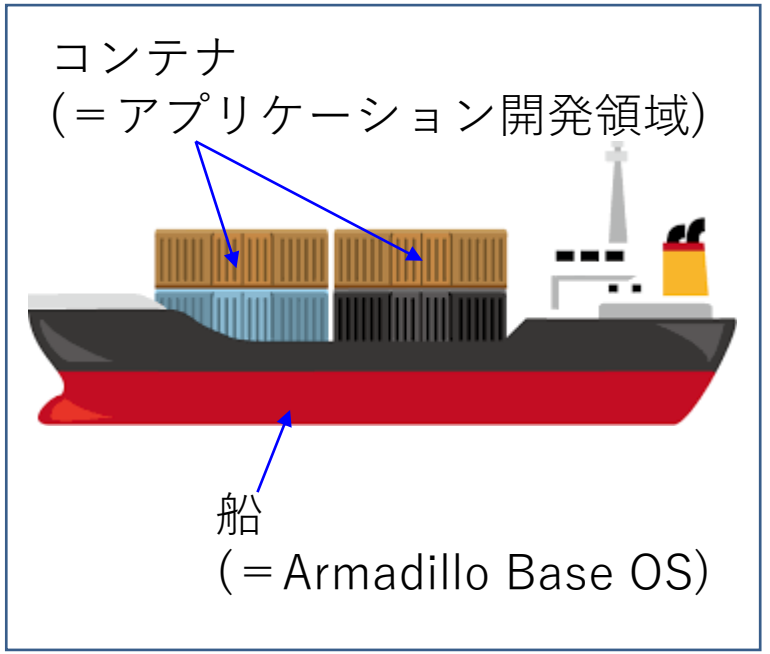
コンテナについて

コンテナとは

コンテナとはカーネルの機能を使用して、ファイルやユーザー、プロセス、ネットワークなどをコンテナ単位で管理する仕組みです。

コンテナの特徴

- コンテナ内の開発環境はユーザーで選択可能
  ⇒DebianやUbuntuといった環境も選択可能
- コンテナはコンテナ外部に対してアクセス制限あり
  ⇒権限を与えない限り、外部へアクセス不可
- コンテナ内の依存関係はコンテナ内でクローズされる
  ⇒コンテナ(アプリ)として持ち出して他の機器に使える
  アップデートにも使用可能
- コンテナを使用する事による処理遅延はほぼ無い
  ⇒プロセスは通常のLinuxと同じ



コンテナの簡単なイメージ

コンテナ(アプリ)開発イメージ

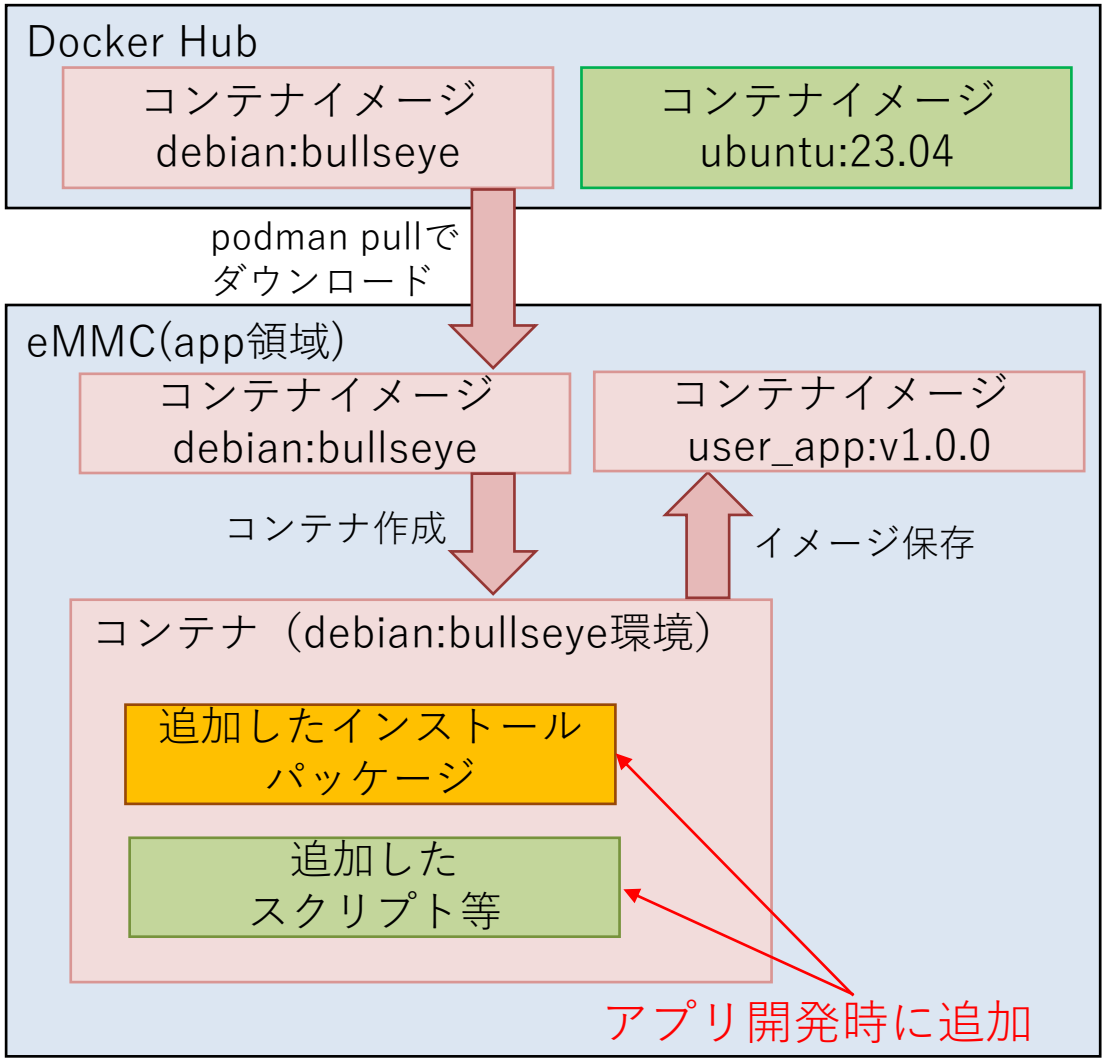
①コンテナを作る際、コンテナイメージという
動作環境テンプレートを使用します。
（Docker Hubについては第3部で説明）

②ここでは例としてdebian:bullseyeのコンテナ
イメージをArmadilloにダウンロードします。

③そのコンテナイメージでコンテナを作成すると
コンテナ内はdebian:bullseyeの環境になります。
(debianのコマンド、パッケージなど使用可能)

④コンテナ内に従来通りアプリケーションを
開発します。

⑤コンテナ(アプリ)が完成したらコンテナイメージ
として保存します。
(開発済みのコンテナとしてアップデートなどで
他の機器にも展開可能)



第１部まとめ

- IoT機器を運用する為に考える事
  - ・長期運用とセキュリティに対応
- Armadillo Base OSの特徴
  - ・アップデート/ロールバック機能
  - ・長期安定運用
  - ・セキュリティ対策
- コンテナについて
  - ・コンテナの概要
  - ・コンテナ(アプリ)開発のイメージ