TRASIO研究技術の全体像

TRASIOが作るセキュリティ技術 (TRV) で今実現できる事、これから実現できそうな事
2021/11/17

セキュアオープンアーキテクチャ・エッジ基盤技術研究組合(TRASIO)/セコム分室
伊藤大輔

この成果は、国立研究開発法人新エネルギー・産業技術総合開発機構（ＮＥＤＯ）の
委託業務（JPNP16007）の結果得られたものです

オープンコミュニティ powered by TRASIO　を設立しました

【設立の背景】

- 国立研究開発法人新エネルギー・産業技術総合開発機構（NEDO）にて、
  「高効率・高速処理を可能とするAIチップ・次世代コンピューティングの技術開発」
  (2018年～2022年）プロジェクト推進中。
- IoT機器を利用したサービスを安全に実現する研究をするのが、TRASIOの課題。
- TRASIOでは、RISC-Vをベースとしたセキュリティシステムを開発。
  本研究開発成果の紹介や試使用、ハンズオン体験を行う場として、
  「オープンコミュニティ powered by TRASIO」を設立した。

これから話すこと

- TRASIOで何をしているのか
- どこまでできているのか
- 今後どうなる予定か

TRASIOで何をしているのか

- RISC-V CPUを含め、安全なIOTサービスを構築するために必要なものの中のいくつかを作ろうとしている
- なぜRISC-V
  - Appleは「自分で使う」ために、「クローズで作る」
    - https://support.apple.com/ja-jp/guide/security/sec59b0b31ff/web
  - Intel,ARMは「みんなで使う」ために「クローズで作る」
    - https://www.intel.co.jp/content/www/jp/ja/architecture-and-technology/software-guard-extensions.html
    - https://www.arm.com/ja/why-arm/technologies/trustzone-for-cortex-m
    - https://www.arm.com/ja/why-arm/technologies/trustzone-for-cortex-a
  - TRASIOは「みんなで使う」ために「オープンで作る」

IOTが安全でなくなる理由（の幾つか）

- デバイスサイドの問題
  - 見られたくないデータを見られてしまう
  - データを改変されてしまう
  - 望まない動作をさせられてしまう
  - 更新が面倒なので古いプログラムがそのまま
- サーバサイドの問題
  - 通信相手が信頼できるデバイスなのかの確認が難しい
- サプライチェーンの問題
  - 機械が作られてゆく過程の安全確保が難しい

解決策

- デバイスサイドの問題
  - 見られたくないデータを見られてしまう
  - データを改変されてしまう
  - 望まない動作をさせられてしまう
  - 更新が面倒なので古いプログラムがそのまま
- サーバサイドの問題
  - 通信相手が信頼できるデバイスなのかの確認が難しい
- サプライチェーンの問題
  - 機械が作られてゆく過程の安全確保が難しい

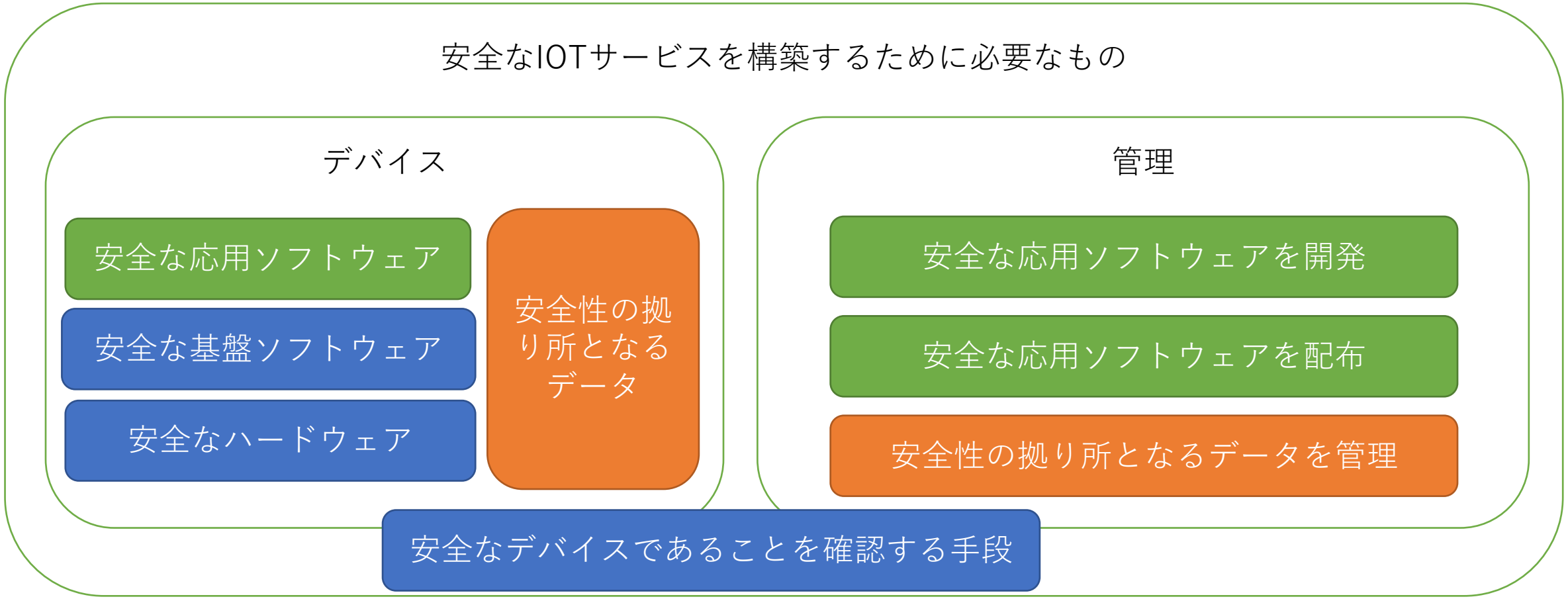
見られたくないデータを見られてしまう／データを改変されてしまう／望まない動作をさせられてしまう → ハードウェアと基盤ソフトで解決

更新が面倒なので古いプログラムがそのまま → ソフト管理の仕組みで解決

通信相手が信頼できるデバイスなのかの確認が難しい → 確認手段を提供で解決

機械が作られてゆく過程の安全確保が難しい → 鍵管理の仕組みで解決

TRASIOが研究していること



TRASIO研究技術の全体像

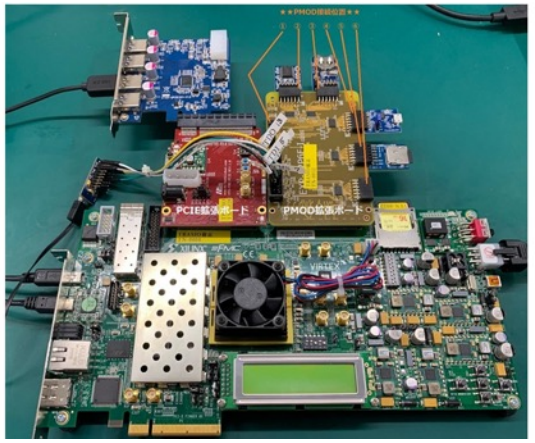
TRASIOが研究していること

- 安全にプログラムを実行できるハードウェア（TRV)
- TRV上で安全にプログラムを実行できる仕組み(TEE)
- 安全なプログラムの開発環境(Global Platform API)
- 安全なプログラムを管理する仕組み(TEEP)
- 色々な「安全」の元になる鍵を管理する仕組み(Key Management)
- TRVで動いていることを外部に宣言する仕組み(Remote Attestation)
- この仕組みを使ってどんなことができるのか
- できることのサンプル(POC)

安全にプログラムを実行できるハードウェア（TRV）

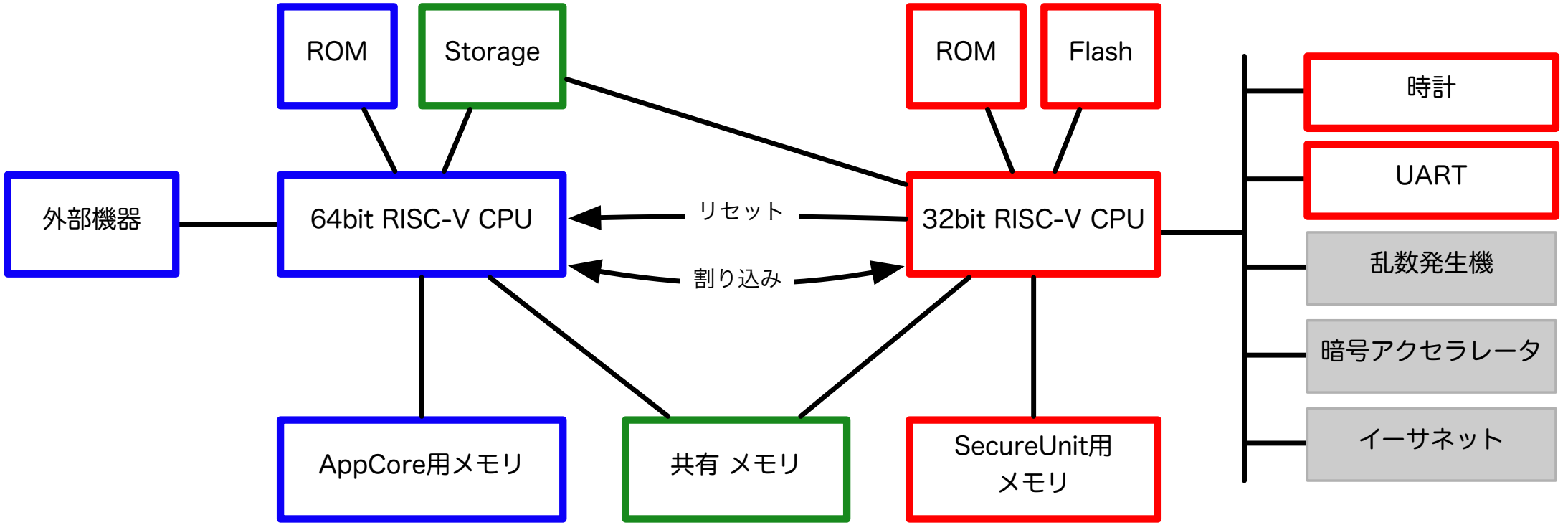
- RISC-V
- AppCore
- SecureUnit
  - Root Of Trust

- **安全にプログラムを実行できるハードウェア（TRV）**
- TRV上で安全にプログラムを実行できる仕組み
- 安全なプログラムの開発環境
- 安全なプログラムを管理する仕組み
- 色々な「安全」の元になる鍵を管理する仕組み
- 正しいCPUで動いていることを外部に宣言する仕組み
- この仕組みを使ってどんなことができるのか
- できることのサンプル



TEE対応CPU・Root Of Trust

TRVの中身


ROM
Storage
ROM
Flash
時計
UART
外部機器
64bit RISC-V CPU
リセット
割り込み
32bit RISC-V CPU
乱数発生機
暗号アクセラレータ
イーサネット
AppCore用メモリ
共有 メモリ
SecureUnit用
メモリ

TRV上で安全にプログラムを実行できる仕組み

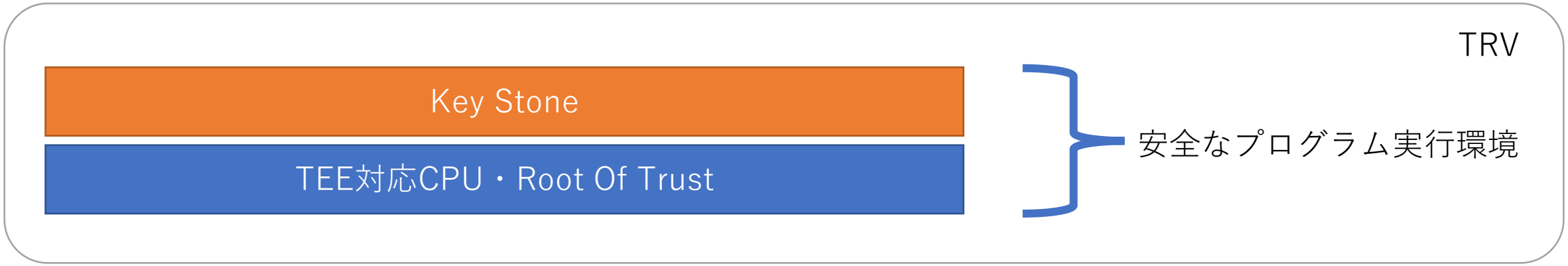
Keystone on RISC-V

An Open Framework for Architecting Trusted Execution Environments

Trust Zone on ARM

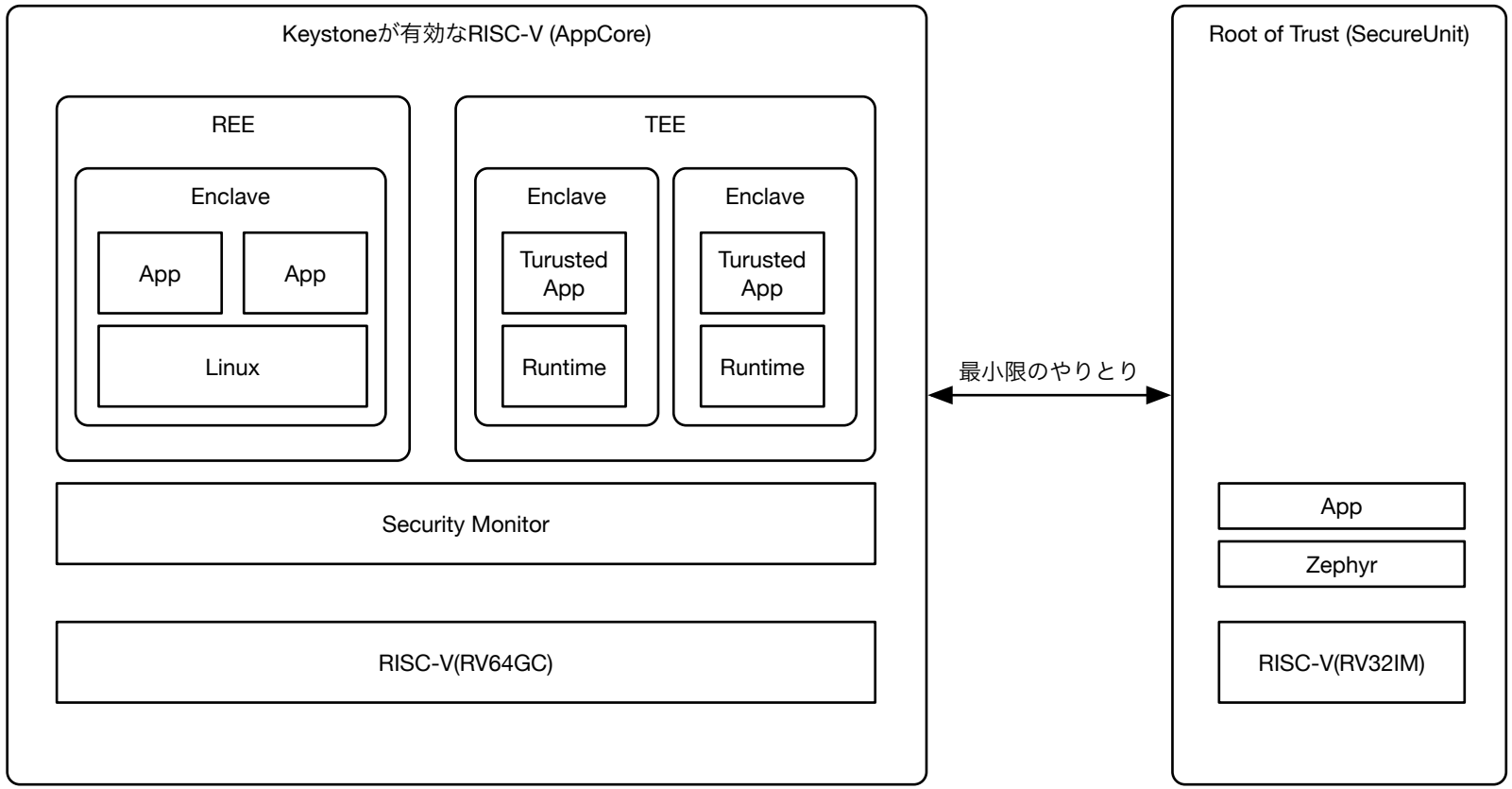
SGX on Intel

- 安全にプログラムを実行できるハードウェア（TRV）
- **TRV上で安全にプログラムを実行できる仕組み**
- 安全なプログラムの開発環境
- 安全なプログラムを管理する仕組み
- 色々な「安全」の元になる鍵を管理する仕組み
- 正しいCPUで動いていることを外部に宣言する仕組み
- この仕組みを使ってどんなことができるのか
- できることのサンプル



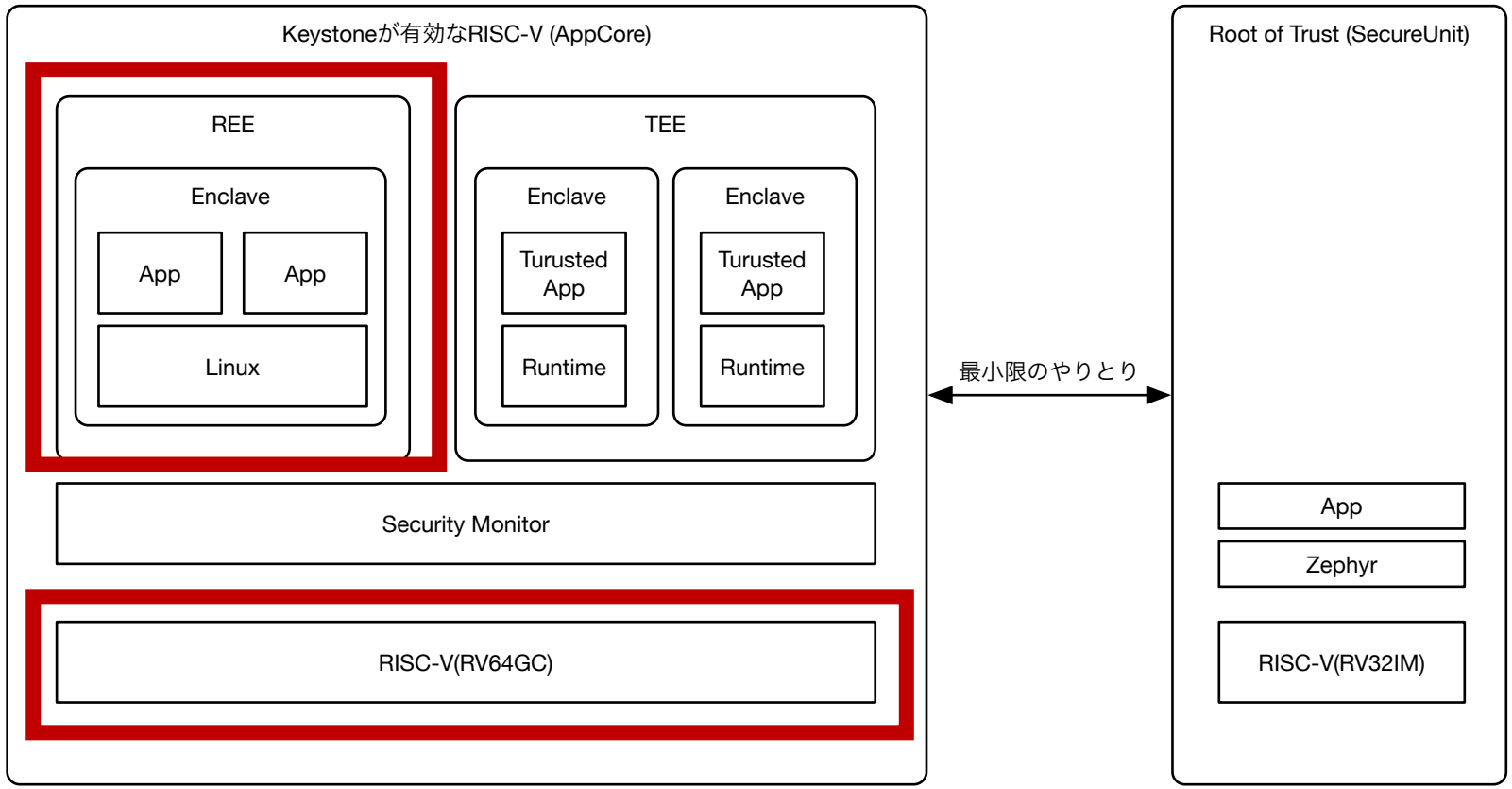
TRASIO研究技術の全体像

Keystoneが有効なRISC-V + SecureUnit = TRV



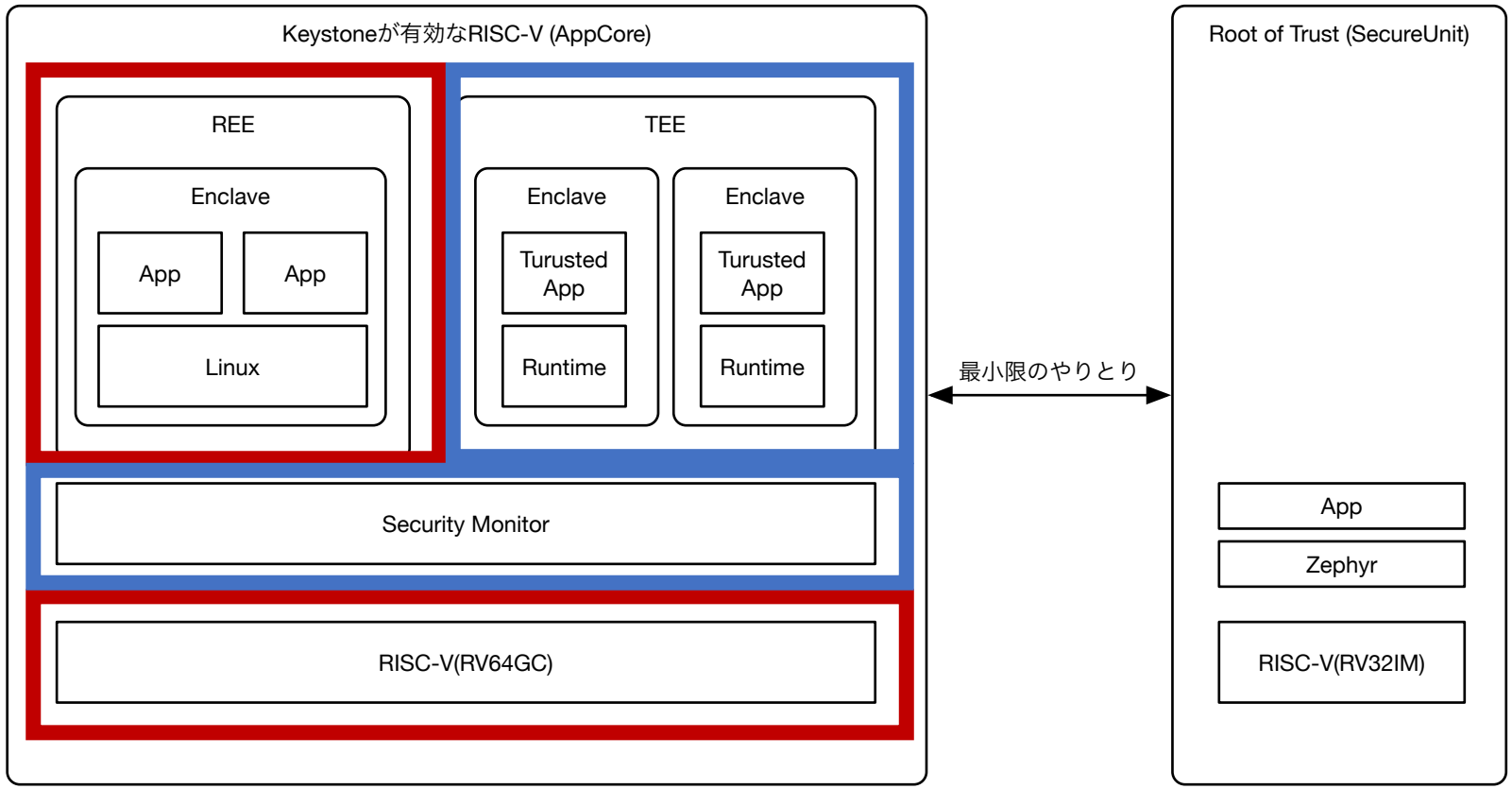
TRASIO研究技術の全体像

Keystoneが有効なRISC-V + SecureUnit = TRV



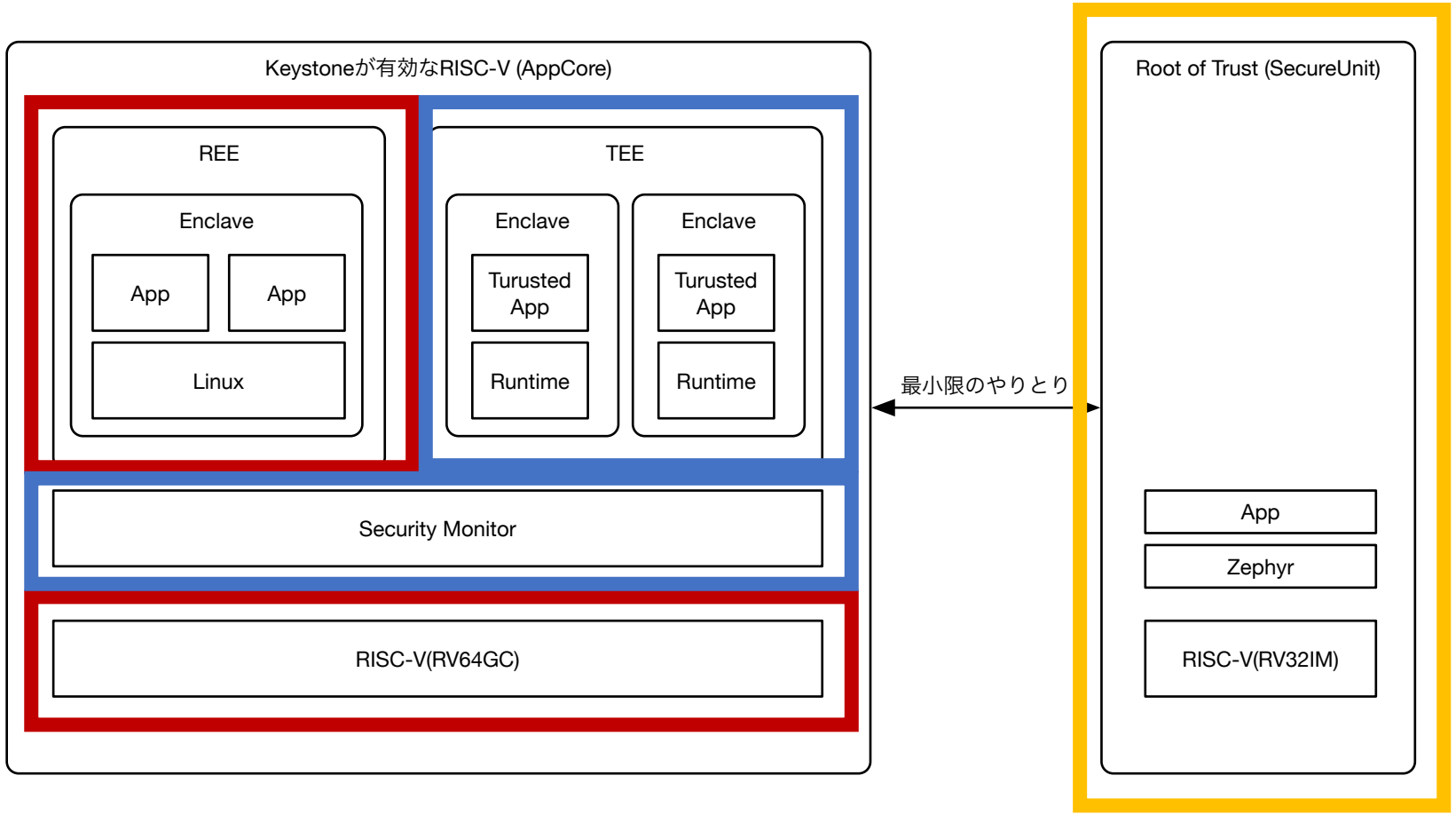
TRASIO研究技術の全体像

Keystoneが有効なRISC-V + SecureUnit = TRV



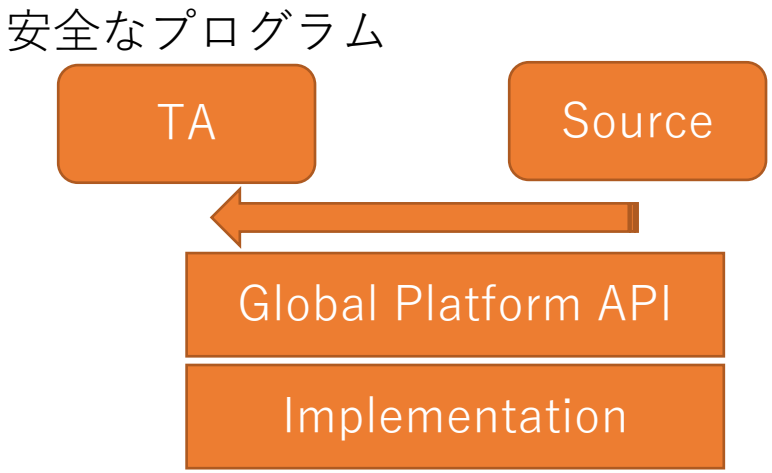
TRASIO研究技術の全体像

Keystoneが有効なRISC-V + SecureUnit = TRV

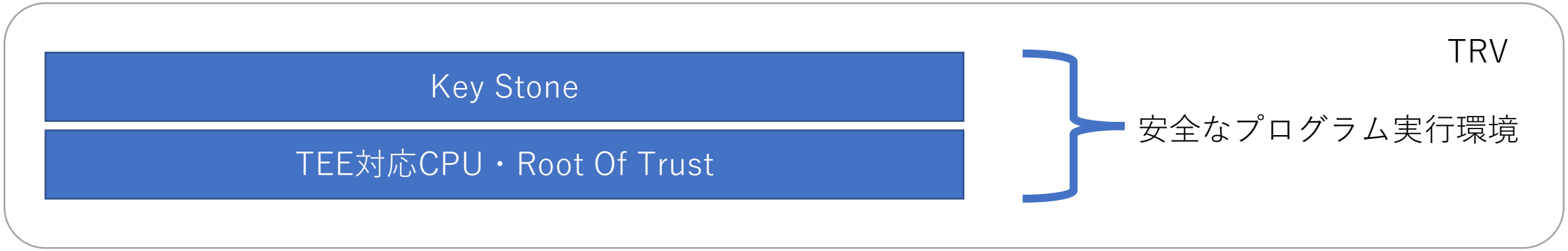


TRASIO研究技術の全体像

安全なプログラムの開発環境+ライブラリ



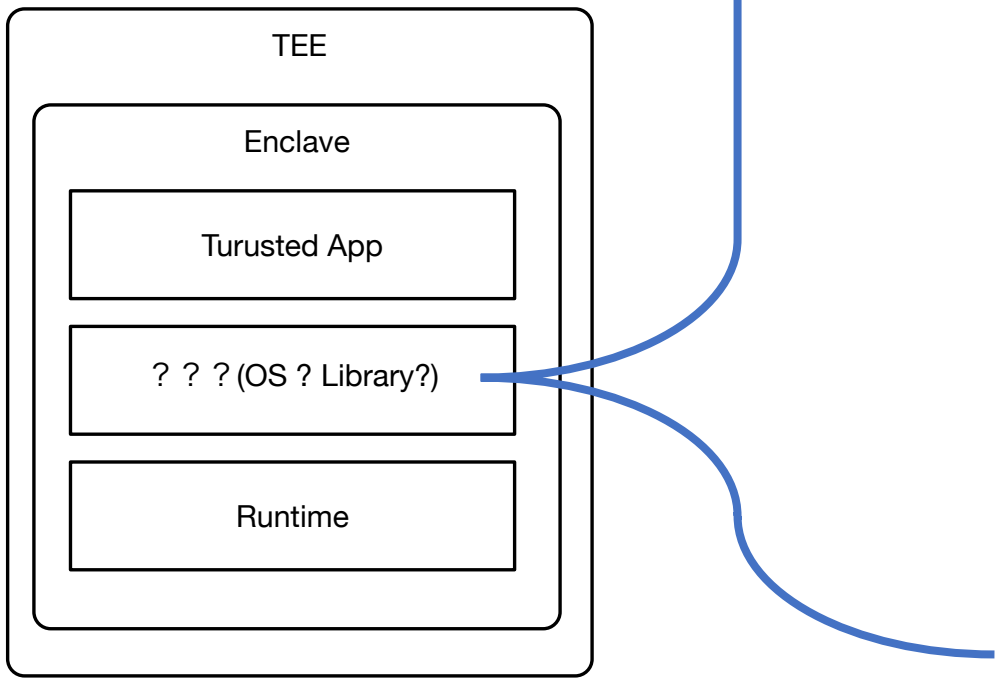
- 安全にプログラムを実行できるハードウェア（TRV）
- TRV上で安全にプログラムを実行できる仕組み
- **安全なプログラムの開発環境**
- 安全なプログラムを管理する仕組み
- 色々な「安全」の元になる鍵を管理する仕組み
- 正しいCPUで動いていることを外部に宣言する仕組み
- この仕組みを使ってどんなことができるのか
- できることのサンプル



TRASIO研究技術の全体像

開発環境は熾烈な争い

- TAを作るのは難しい



Intel SGX用

- Intel SGX SDK
- Haven[OSDI14]
- SCONE[OSDI16]
- Glamdring[USENIX17]
- SGX-LKL (Linux Kernel Library) [arXiv19]
- VC3[IEEE SP15]
- OpenEnclave [Microsoft]

ARM TZ用

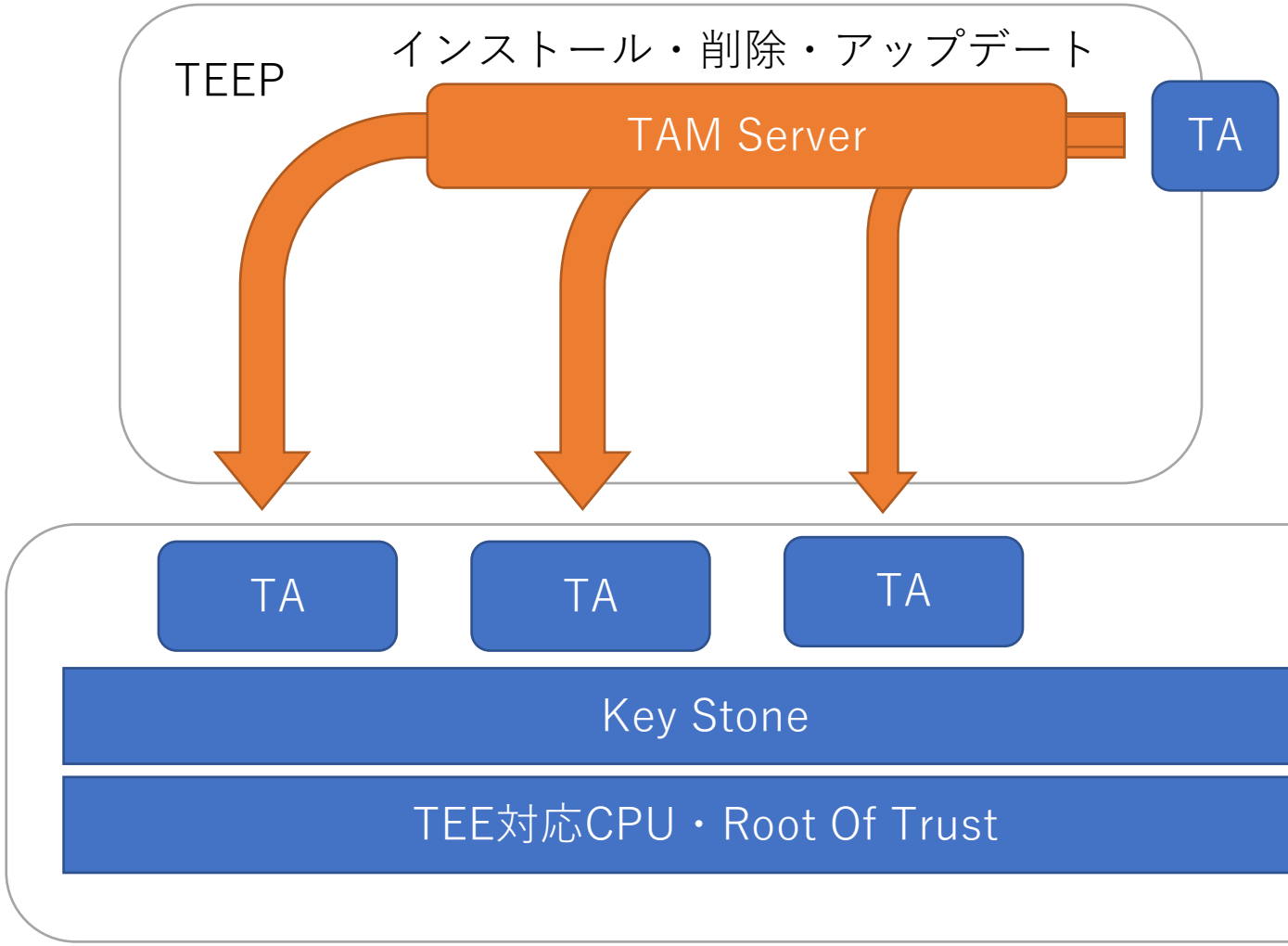
- OP-TEE (Linaro) https://github.com/OP-TEE
- Open-TEE (Aalto University[TrustCom15]) https://open-tee.github.io/
- Trusty (Google) https://source.android.com/security/trusty/index.html
- SierraTEE (Sierra) https://www.sierraware.com/open-source-ARM-TrustZone.html

- Apple's Secure Enclave
- Qualcomm's QTEE, ex. QSEE https://www.qualcomm.com/solutions/mobile-computing/features/security
- Samsung's Knox https://www.samsungknox.com/en
- Samsung's Teegris http://developer.samsung.com/teegris
- Trustonic's Kinibi OS, ex. Mobicore/t-base/G&D
- Huawei's TrustedCore
- Nvidia's TLK (Trusted Little Kernel)

TRASIO研究技術の全体像

TRASIO 須崎さんのスライドから借用
http://www.ipsj.or.jp/sig/os/index.php?plugin=attach&pcmd=open&file=ComSys2020-Suzaki.pdf&refer=ComSys2020

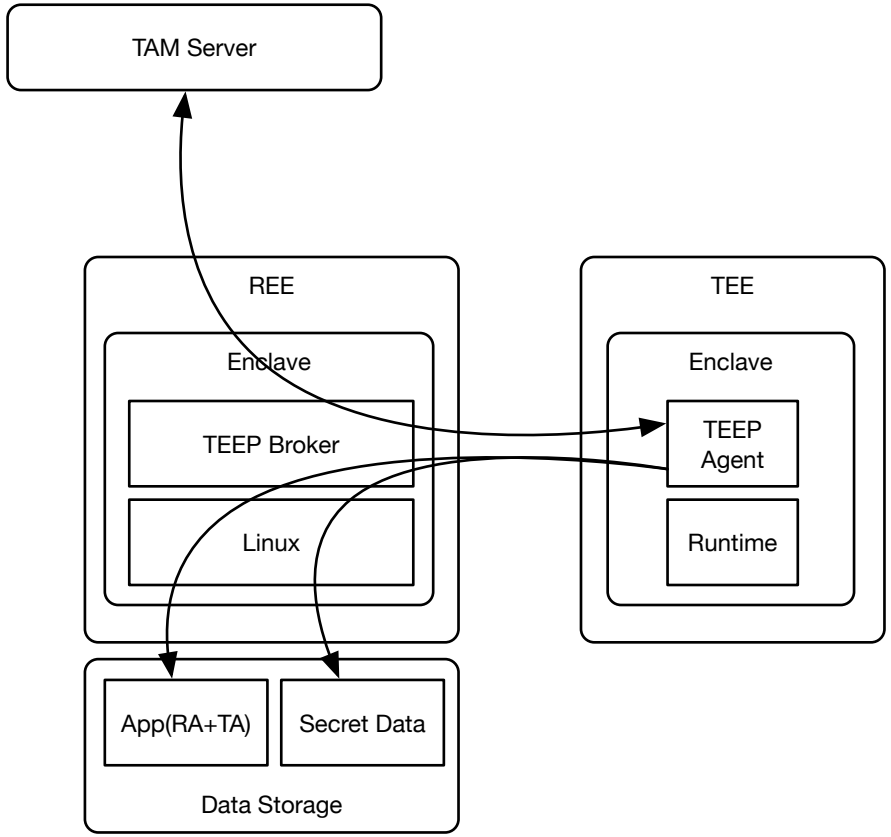
安全なプログラムを管理する仕組み



- 安全にプログラムを実行できるハードウェア（TRV）
- TRV上で安全にプログラムを実行できる仕組み
- 安全なプログラムの開発環境
- **安全なプログラムを管理する仕組み**
- 色々な「安全」の元になる鍵を管理する仕組み
- 正しいCPUで動いていることを外部に宣言する仕組み
- この仕組みを使ってどんなことができるのか
- できることのサンプル

TRASIO研究技術の全体像

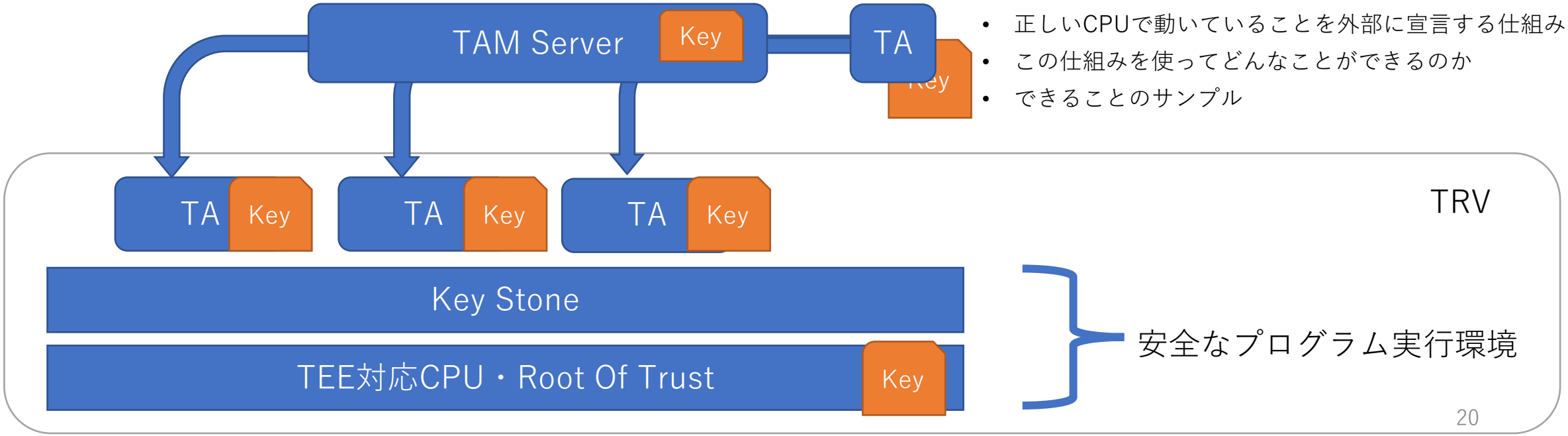
Trusted Execution Environment Provisioning



- TAM(Trusted Application Manager) Server
  - リモート側
  - アプリケーションを配布する
- TEEP Broker
  - TEEP AgentとTAM Serverの通信を中継する
  - TEEP Agentからの指示でデータを書く
  - データに関しては中継するだけ
- TEEP Agent
  - インストールされているAPPの一覧
  - アップデート
  - インストール
  - 削除
  - その他いろいろ

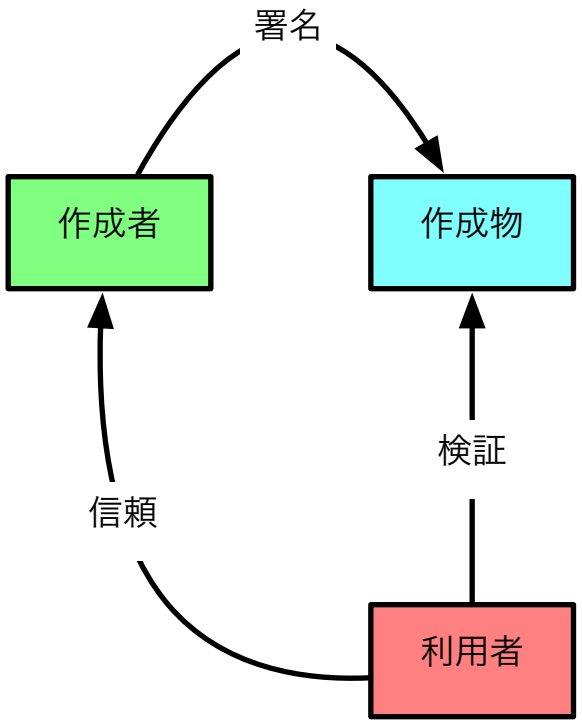
色々な「安全」の元になる鍵を管理する仕組み

- 安全にプログラムを実行できるハードウェア（TRV）
- TRV上で安全にプログラムを実行できる仕組み
- 安全なプログラムの開発環境
- 安全なプログラムを管理する仕組み
- **色々な「安全」の元になる鍵を管理する仕組み**
- 正しいCPUで動いていることを外部に宣言する仕組み
- この仕組みを使ってどんなことができるのか
- できることのサンプル

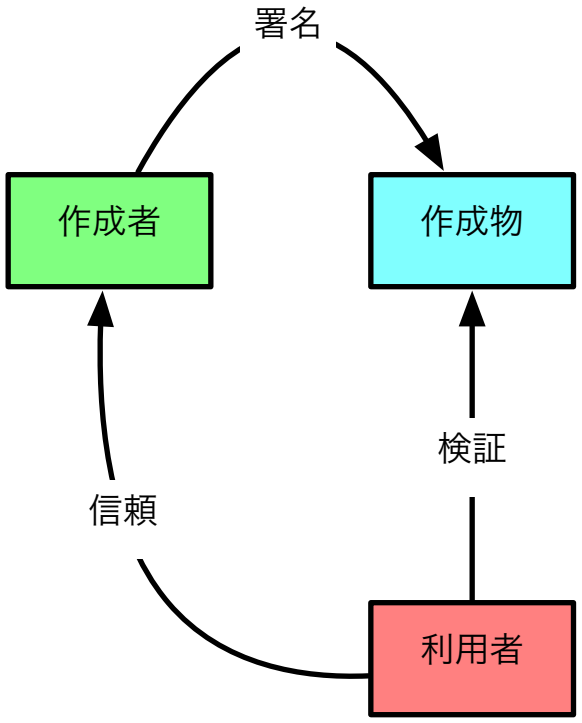


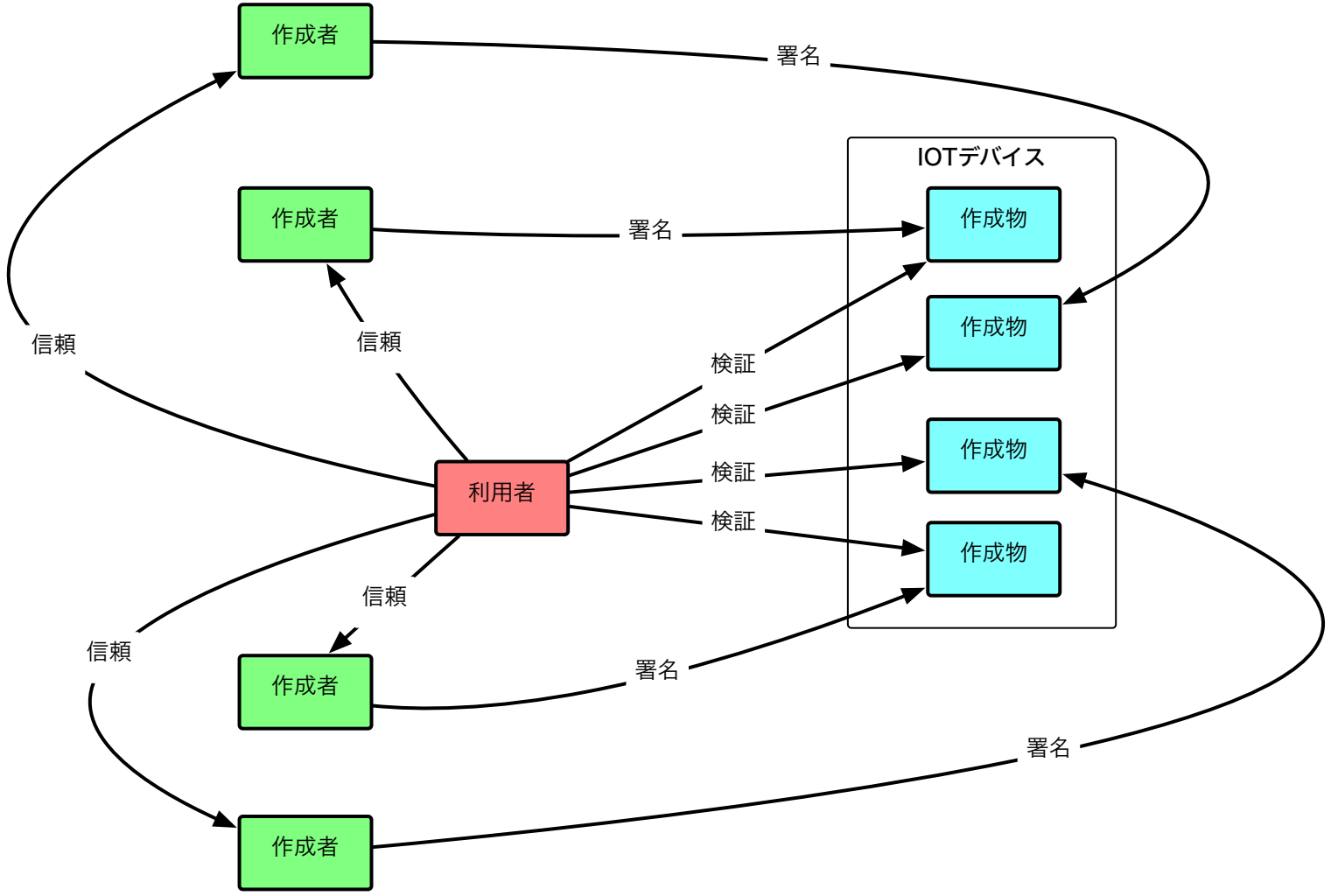
TRASIO研究技術の全体像

製品を信頼するとは？



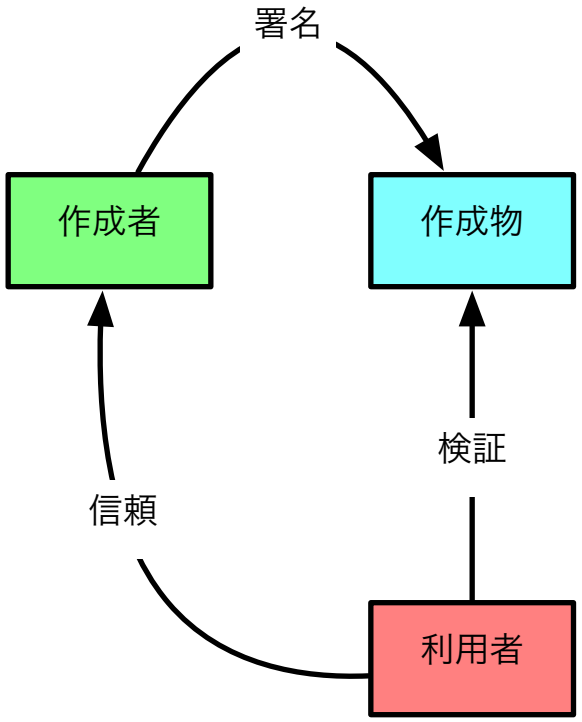
たくさんの人で作った製品を信頼するのは大変

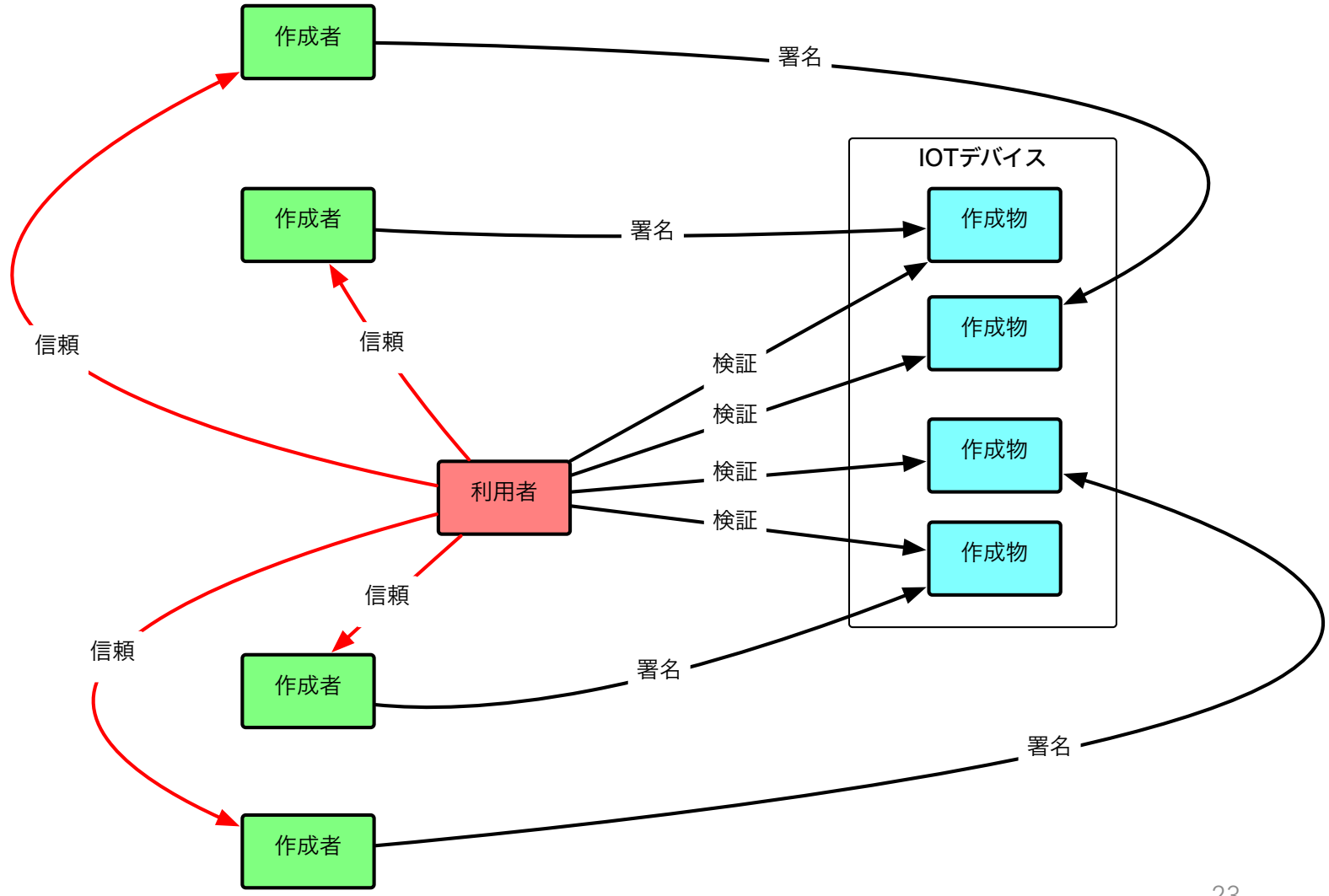




TRASIO研究技術の全体像

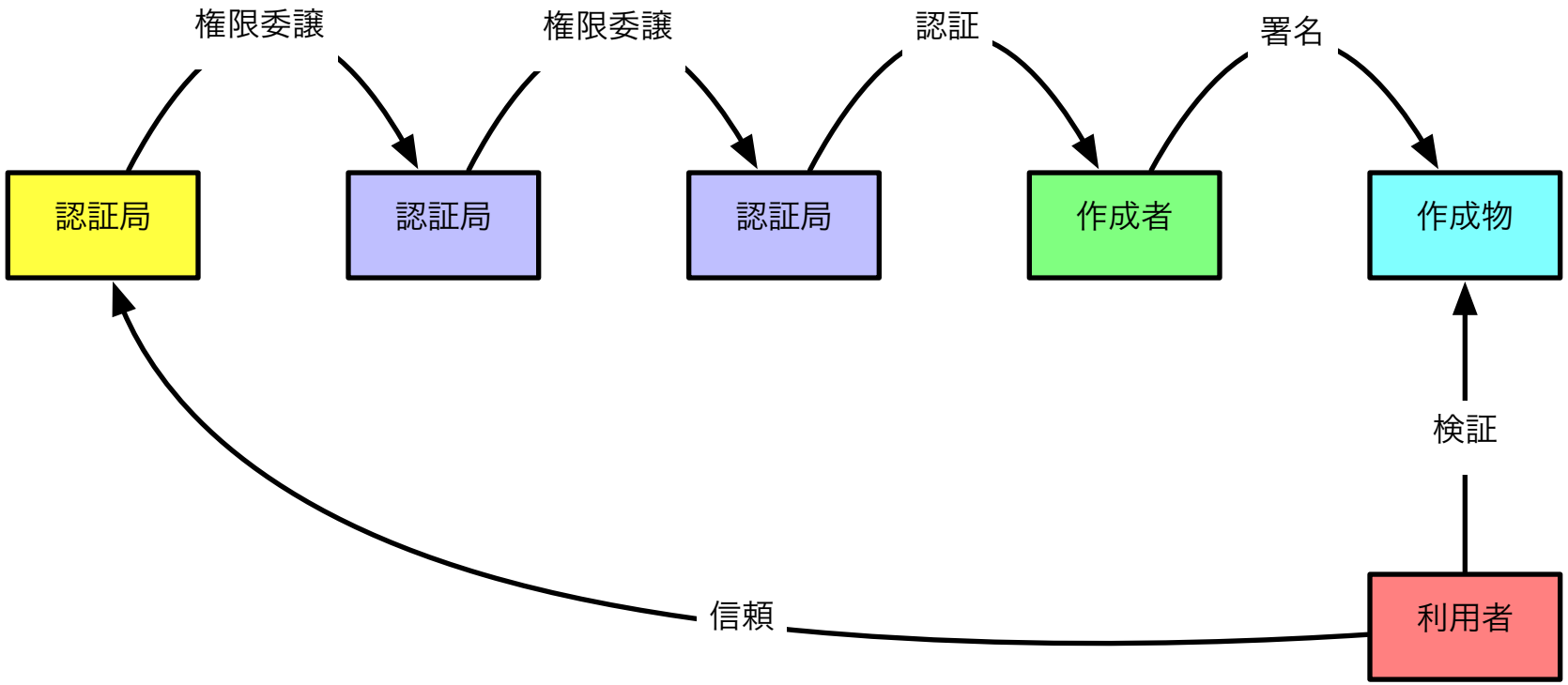
色々な人を信頼しなければならない



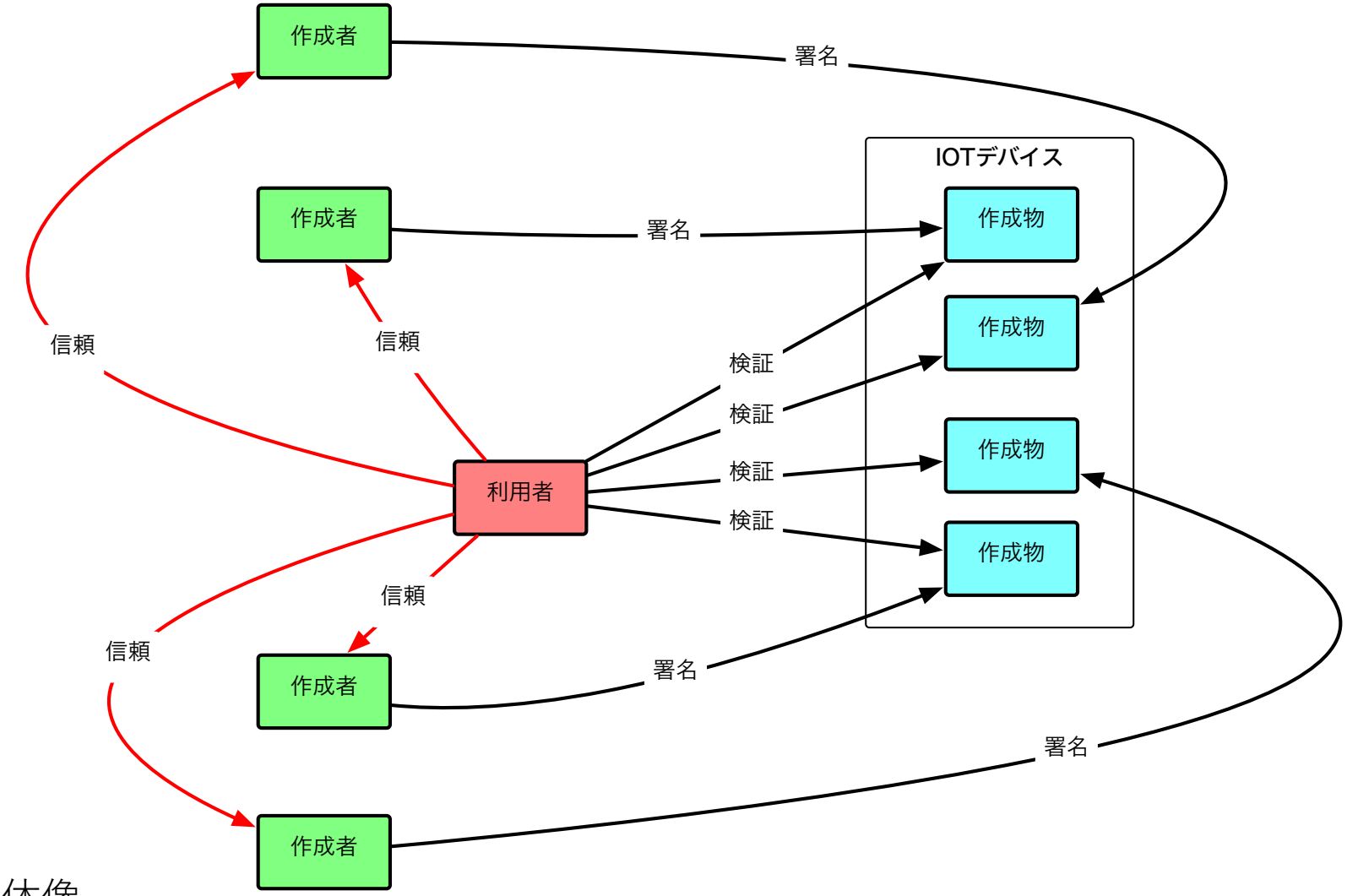


TRASIO研究技術の全体像

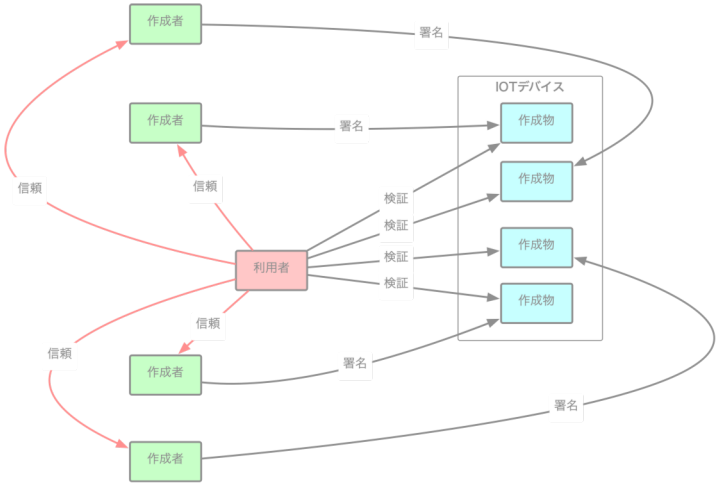
信頼の連鎖(PKI)

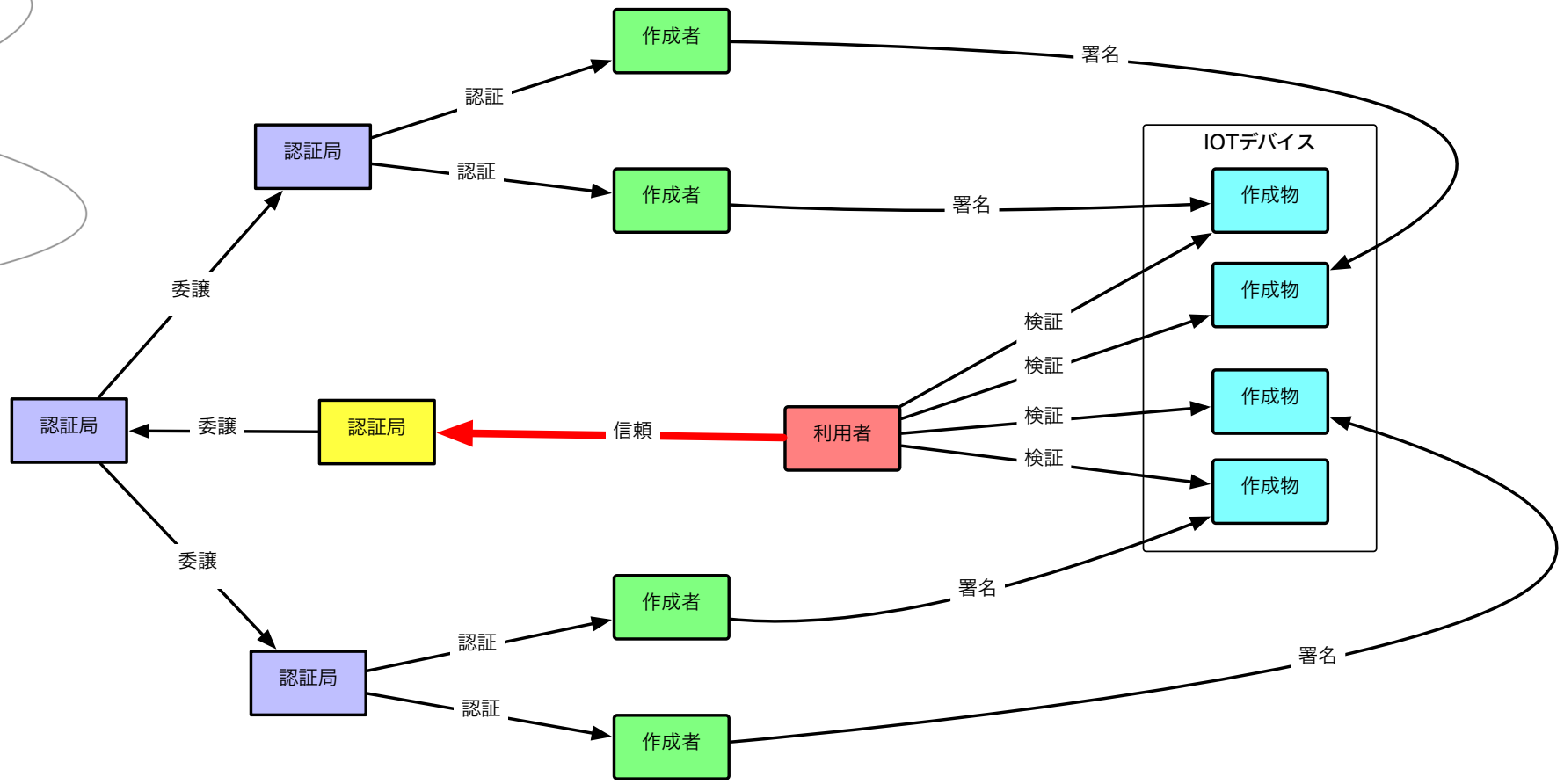


信頼の連鎖導入前

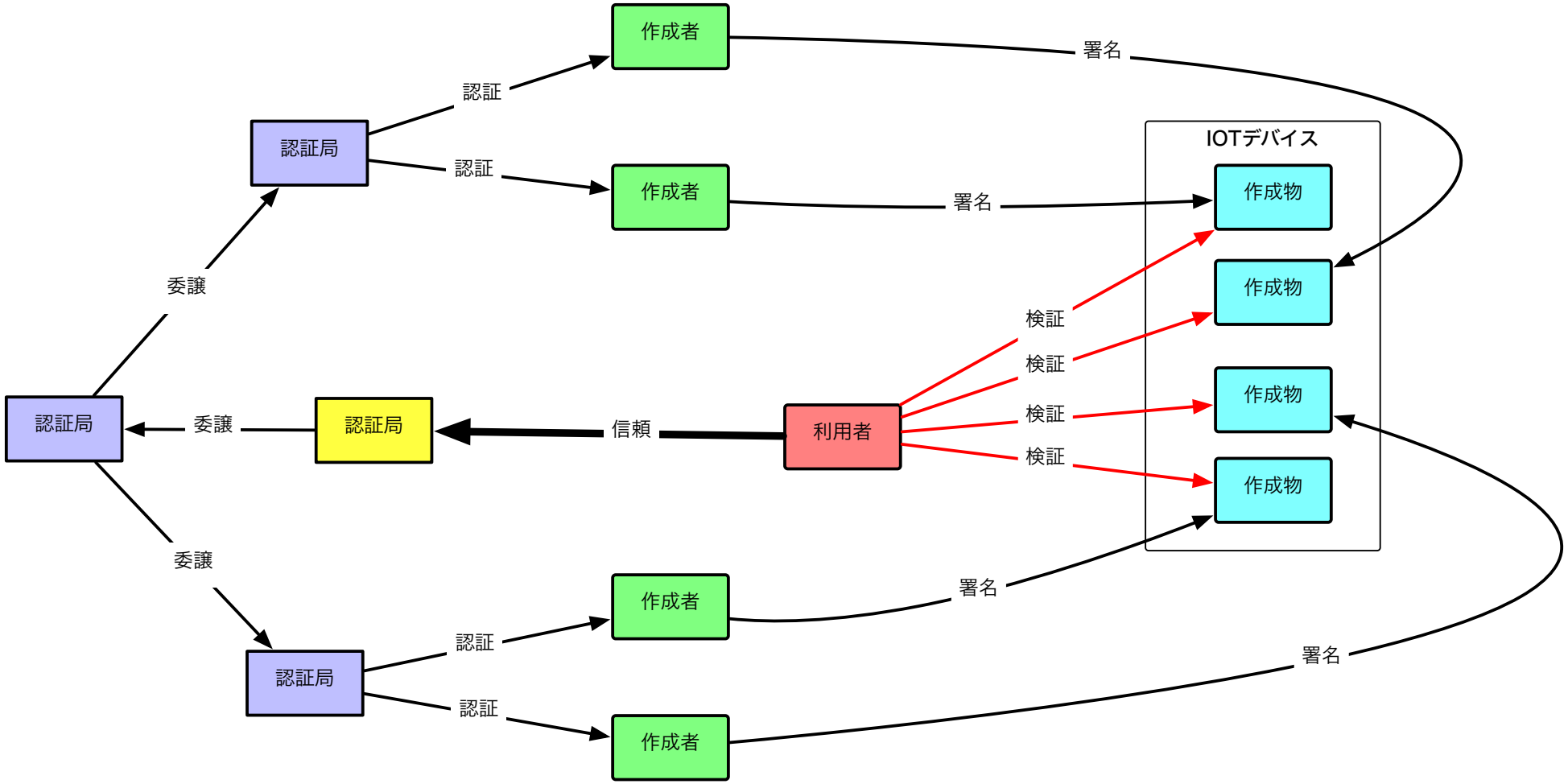

作成者
署名
IOTデバイス
作成者
署名
作成物
作成物
信頼
信頼
検証
検証
検証
検証
作成物
利用者
信頼
信頼
作成者
署名
署名
作成者

信頼の連鎖導入後


作成者
署名
IOTデバイス
作成物
信頼
検証
利用者

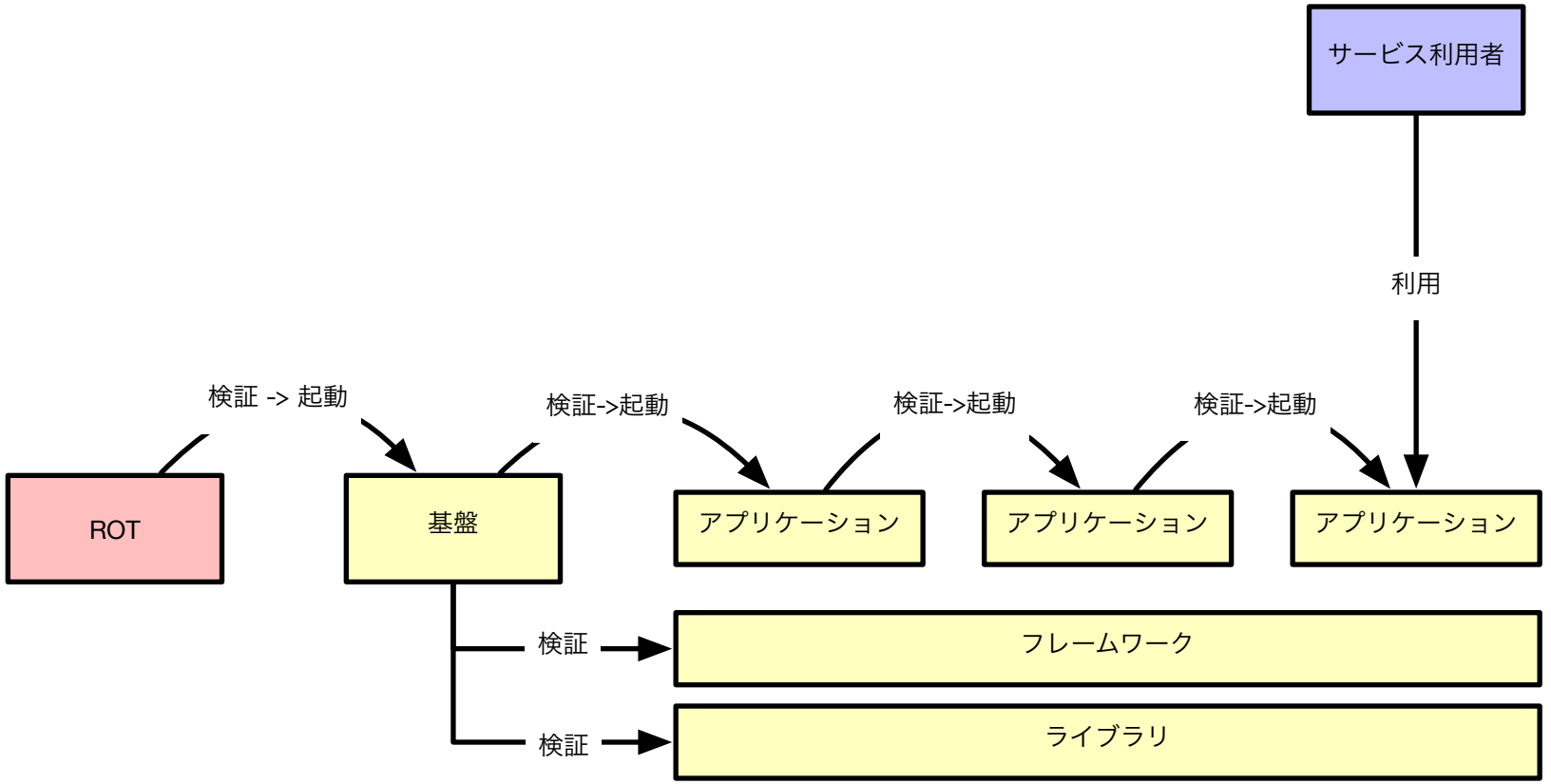

作成者
署名
認証局
認証
委譲
IOTデバイス
作成物
検証
信頼
利用者

色々なものを検証しなければならない

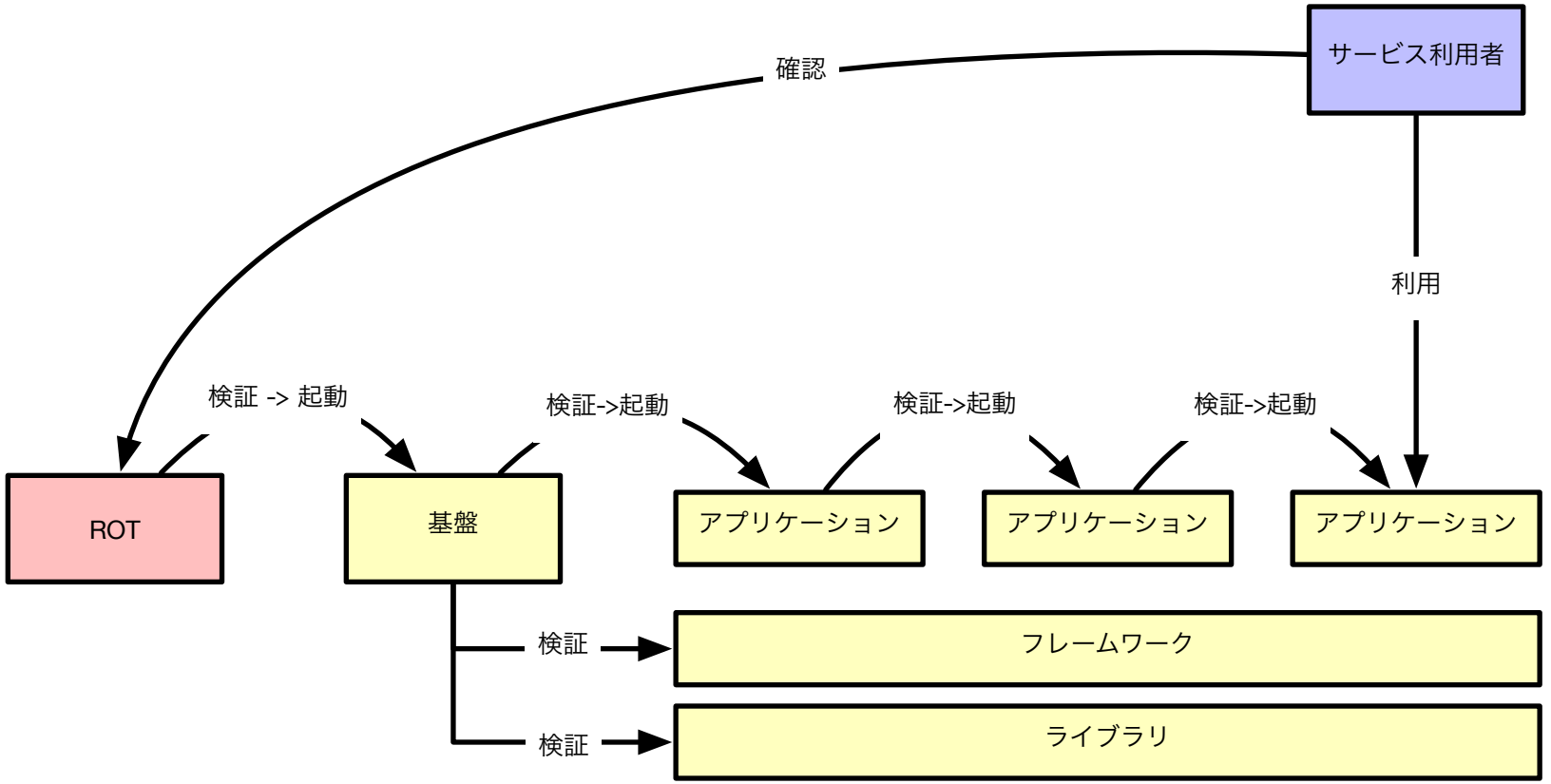


TRASIO研究技術の全体像

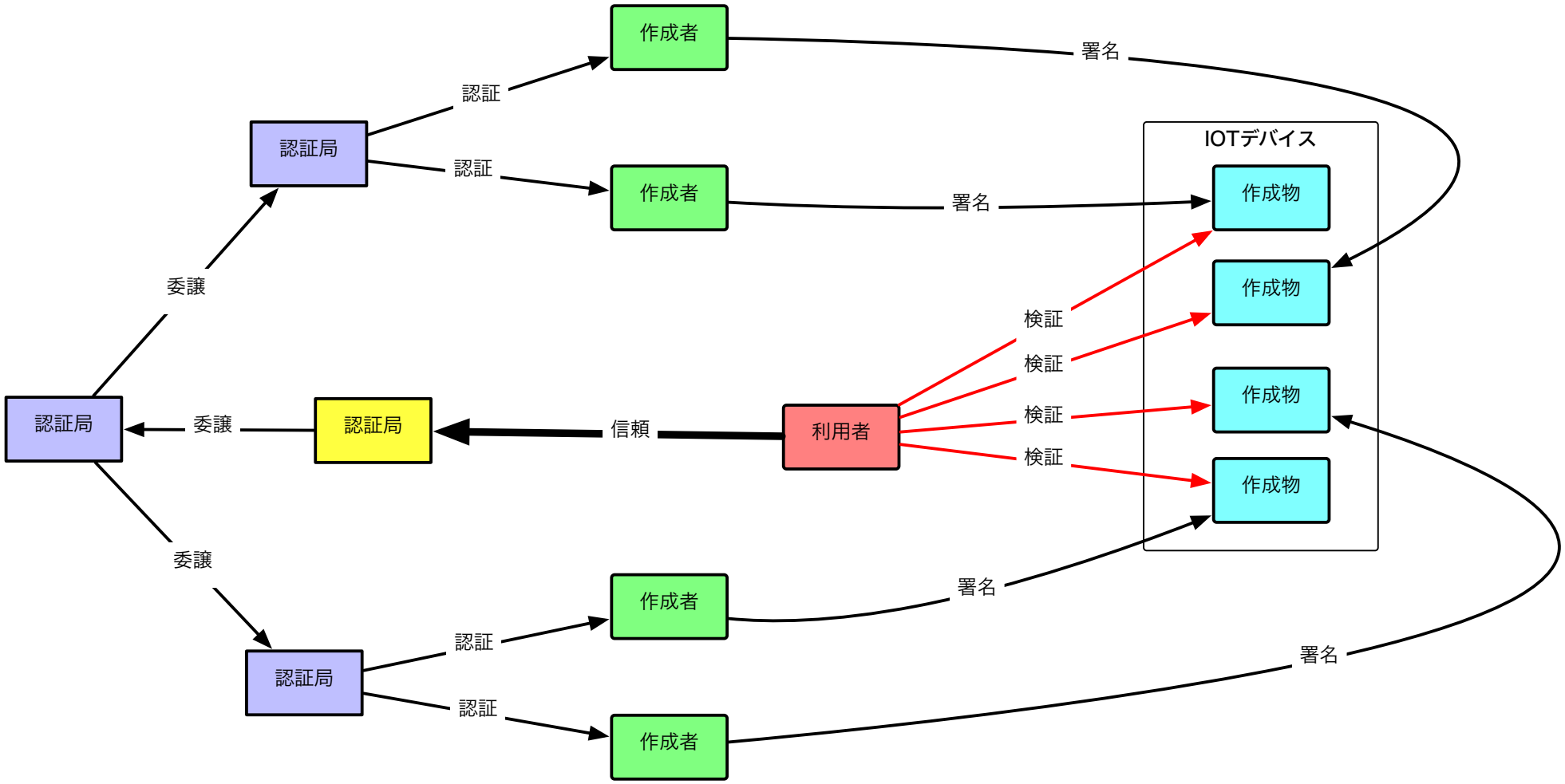
Root Of Trustから連続した検証



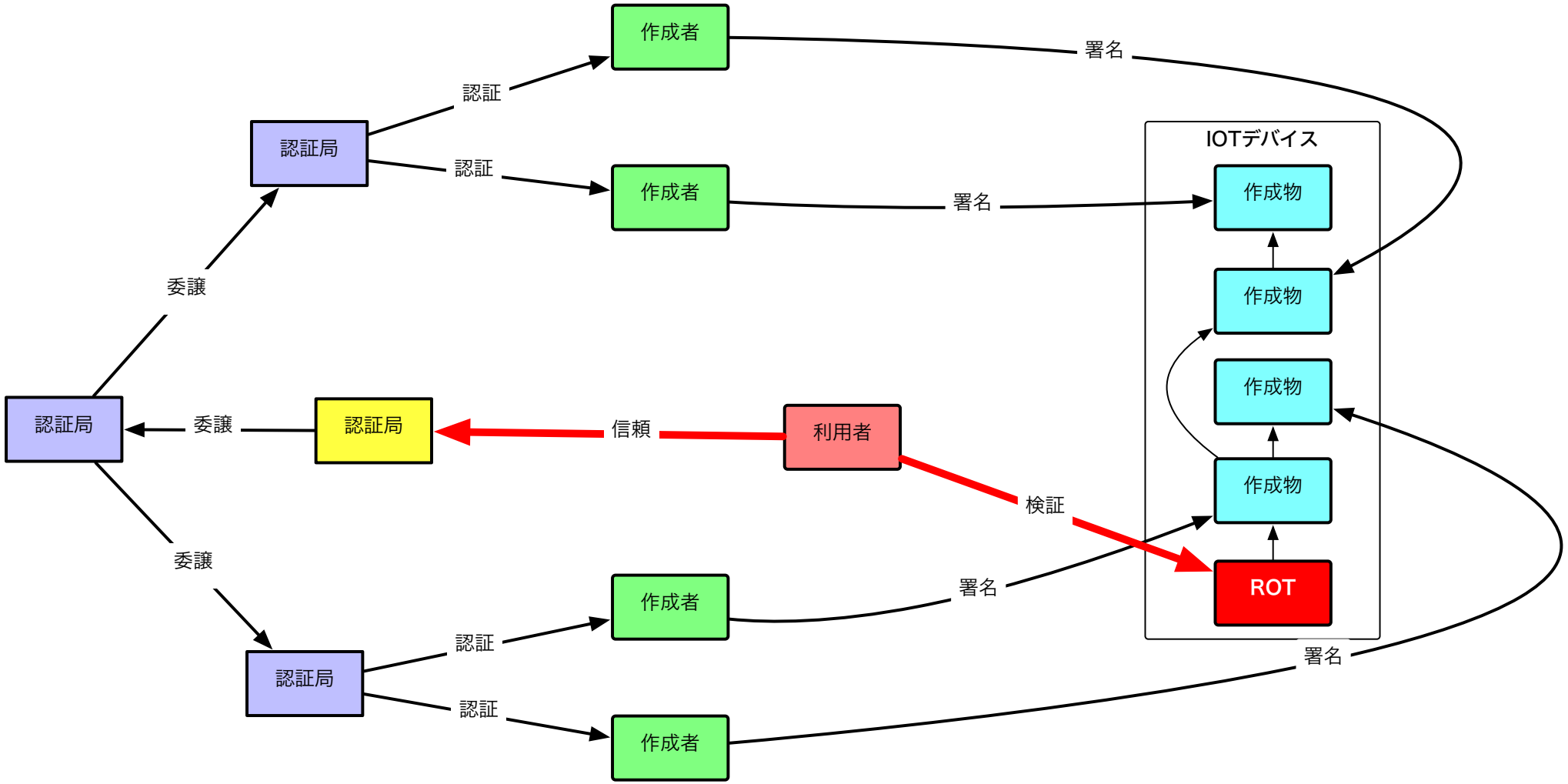
信じられる理由（起動時）


サービス利用者
確認
利用
検証 -> 起動
検証->起動
検証->起動
検証->起動
ROT
基盤
アプリケーション
アプリケーション
アプリケーション
検証
フレームワーク
検証
ライブラリ

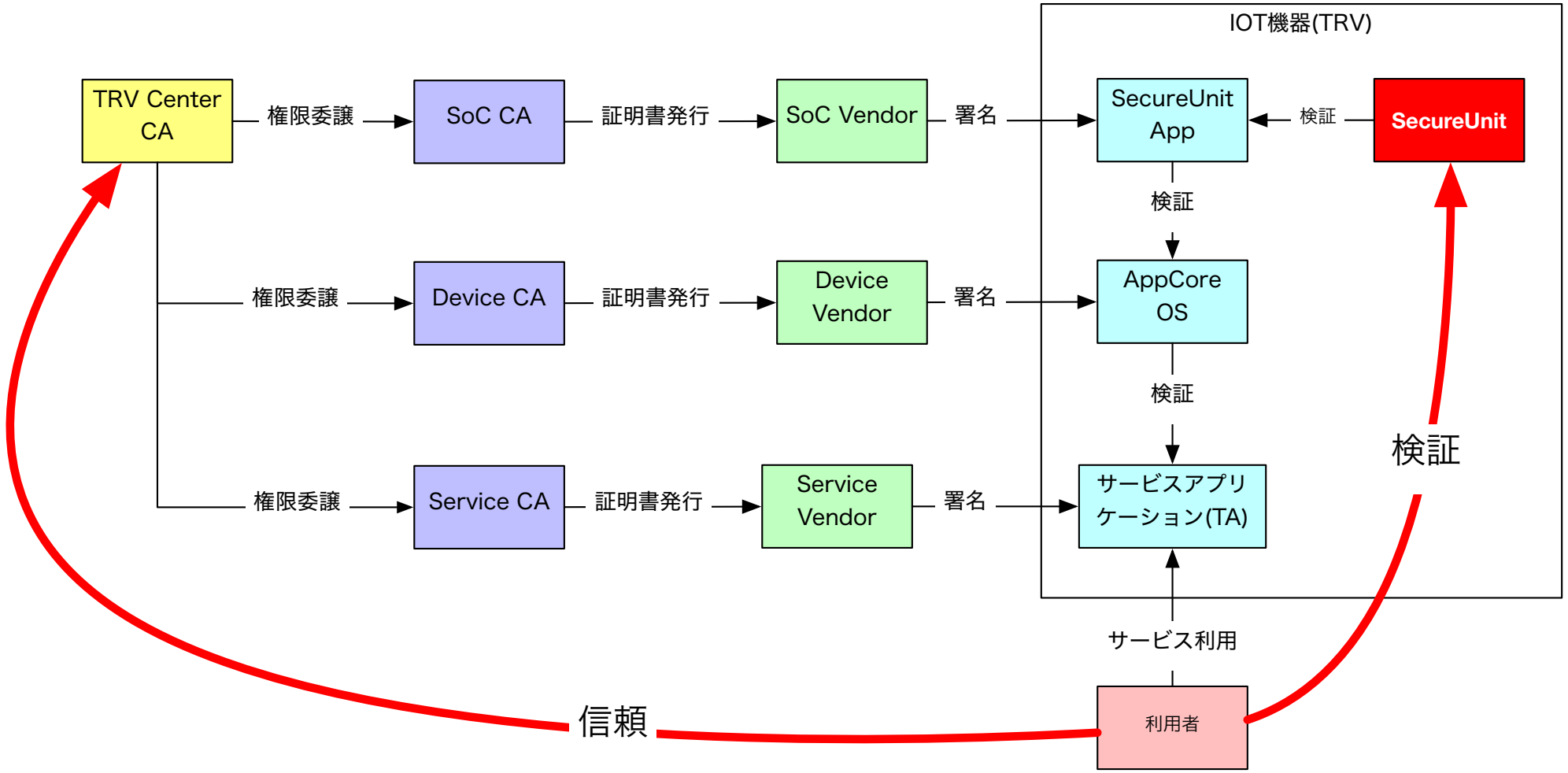
検証の連鎖導入前


作成者
署名
認証
認証局
認証
作成者
署名
IOTデバイス
作成物
委譲
作成物
検証
検証
認証局
委譲
認証局
信頼
利用者
検証
検証
作成物
作成物
委譲
署名
作成者
認証
認証局
認証
署名
作成者

検証の連鎖導入後

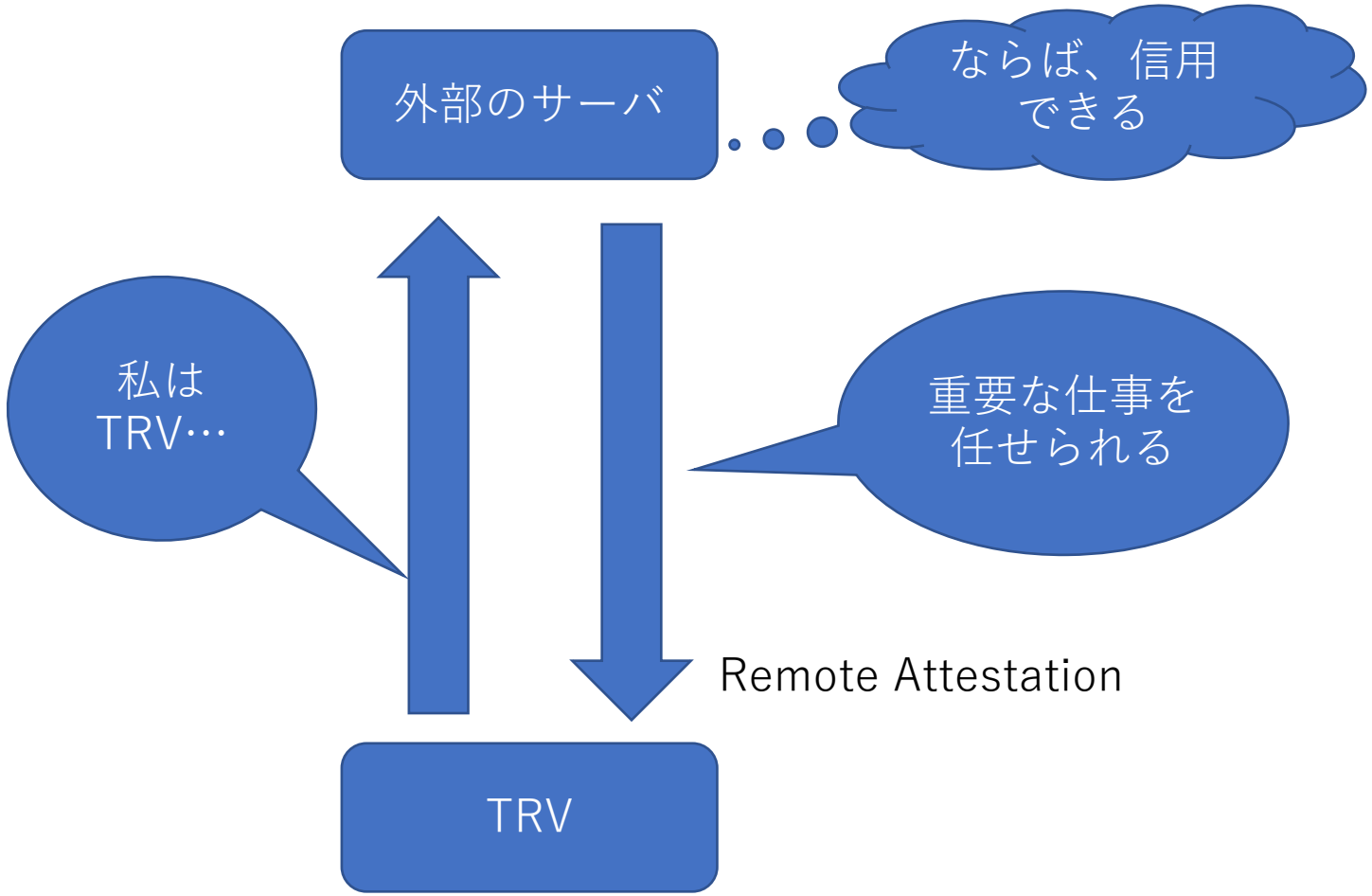

作成者
署名
認証
認証局
認証
作成者
署名
IOTデバイス
作成物
委譲
作成物
作成物
認証局
委譲
認証局
信頼
利用者
検証
作成物
ROT
委譲
署名
作成者
認証局
認証
署名
認証
作成者

TRASIOモデル



TRASIO研究技術の全体像

正しいCPUで動いていることを外部に宣言する仕組み



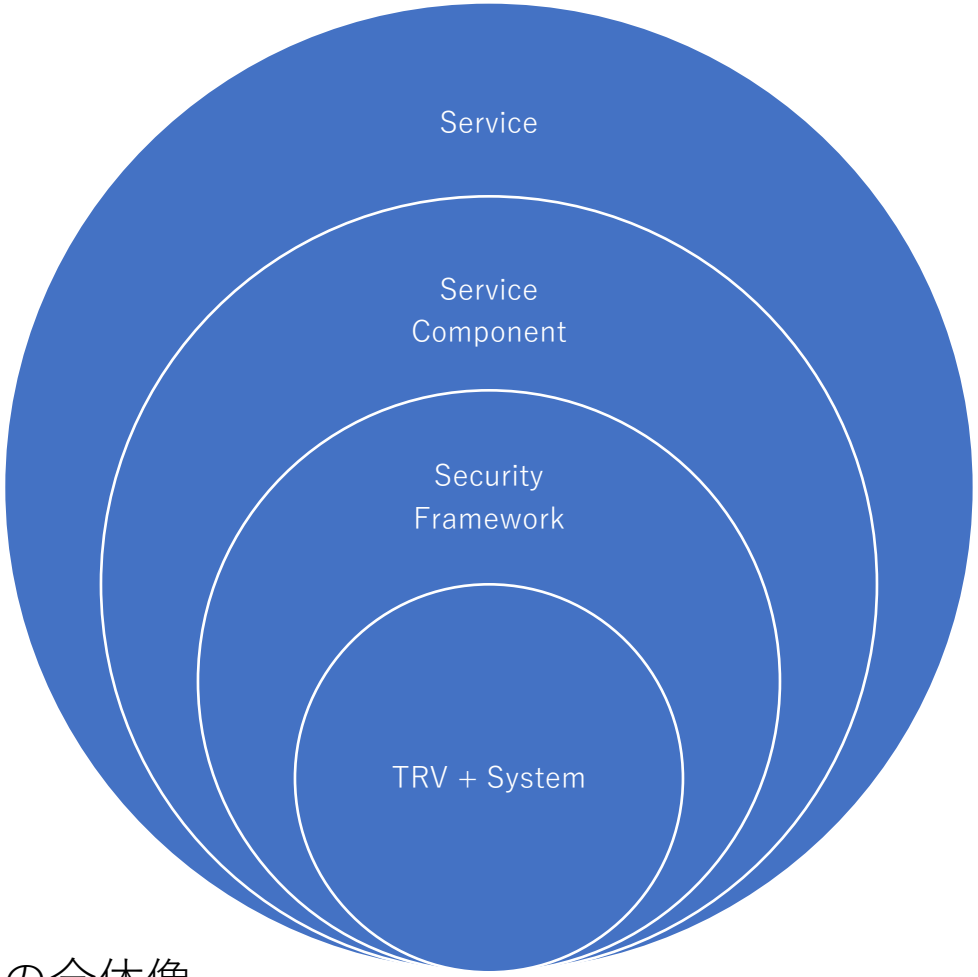
- 安全にプログラムを実行できるハードウェア（TRV）
- TRV上で安全にプログラムを実行できる仕組み
- 安全なプログラムの開発環境
- 安全なプログラムを管理する仕組み
- 色々な「安全」の元になる鍵を管理する仕組み
- **正しいCPUで動いていることを外部に宣言する仕組み**
- この仕組みを使ってどんなことができるのか
- できることのサンプル

TRASIO研究技術の全体像

Remote Attestation

- TLSのクライアント認証と同じようなもの
- Serverから送られてきたNonceを含むClientが生成したクレームにSecureUnitに格納されたデバイス固有の鍵で署名して、サーバに送る
- クレームに何を含めるべきか？それはいろいろ
  - ファームウェアのハッシュ
  - 設定データのハッシュ
  - デバイスツリー

この仕組みを使ってどんなことができるのか



- 安全にプログラムを実行できるハードウェア（TRV）
- TRV上で安全にプログラムを実行できる仕組み
- 安全なプログラムの開発環境
- 安全なプログラムを管理する仕組み
- 色々な「安全」の元になる鍵を管理する仕組み
- 正しいCPUで動いていることを外部に宣言する仕組み
- **この仕組みを使ってどんなことができるのか**
- できることのサンプル

TRASIO研究技術の全体像

できることのサンプル

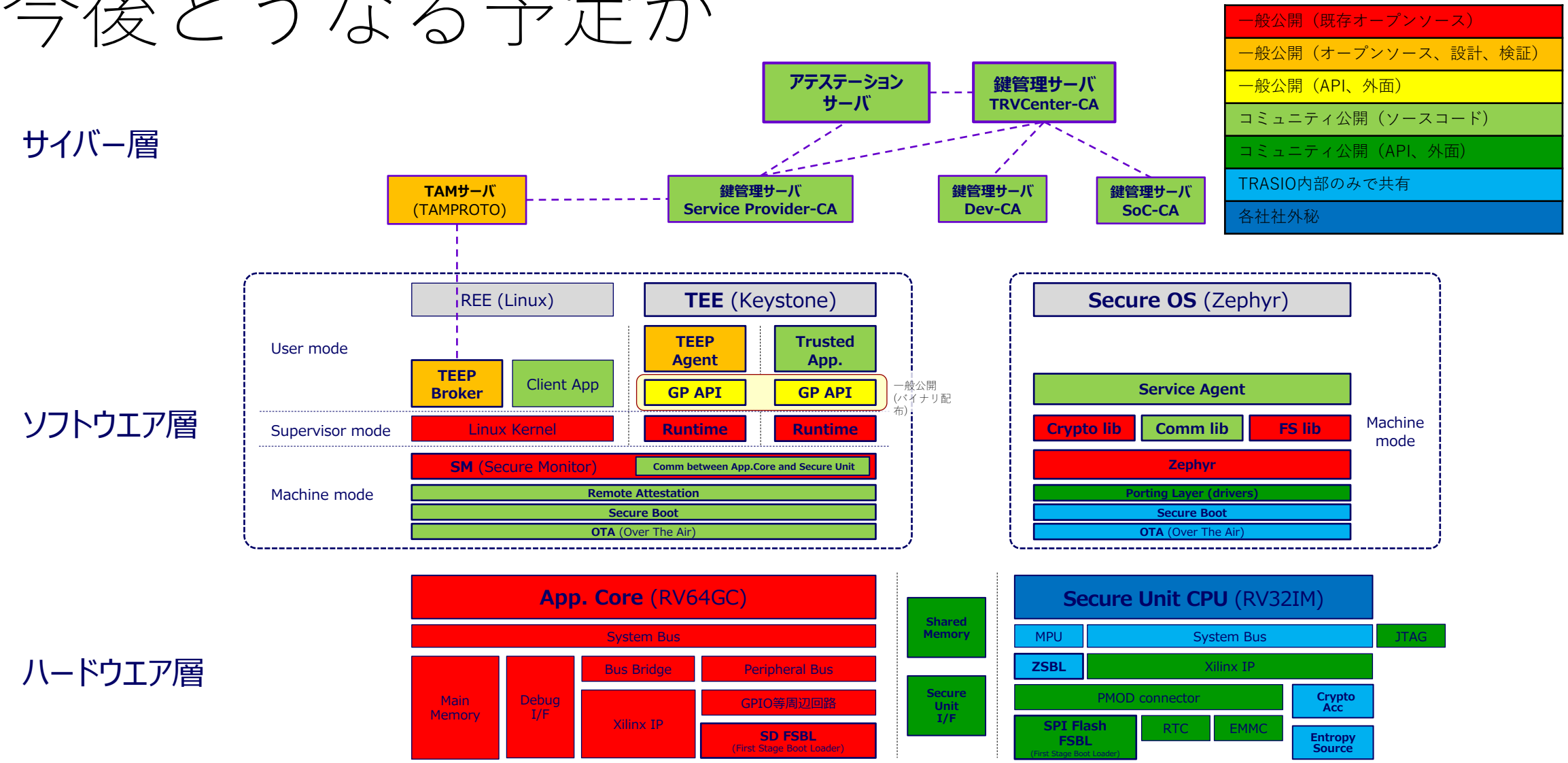
- 安全にプログラムを実行できるハードウェア（TRV）
- TRV上で安全にプログラムを実行できる仕組み
- 安全なプログラムの開発環境
- 安全なプログラムを管理する仕組み
- 色々な「安全」の元になる鍵を管理する仕組み
- 正しいCPUで動いていることを外部に宣言する仕組み
- この仕組みを使ってどんなことができるのか
- **できることのサンプル**

TRASIO研究技術の全体像

どこまでできているのか

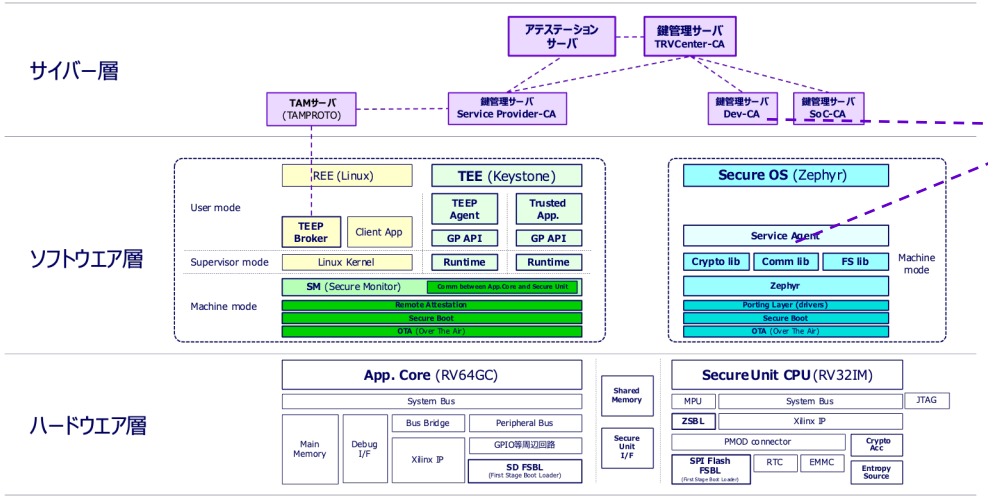
- 研究段階
- 性能追求していない
- ほとんどの機能は動く
- 細かい部分を実装中

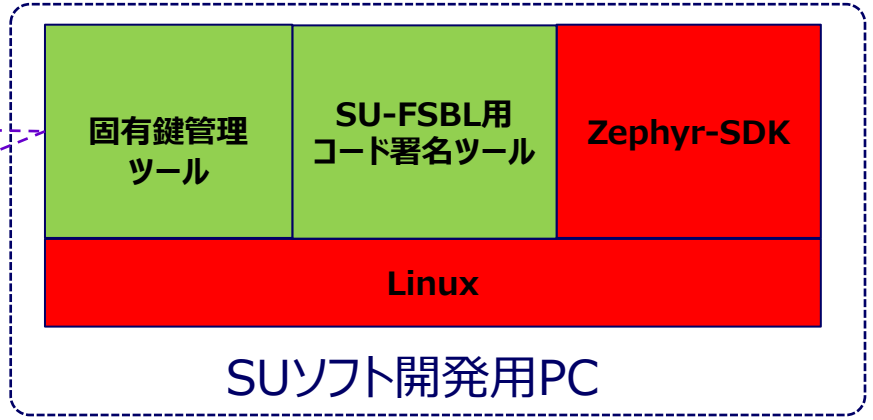
今後どうなる予定か



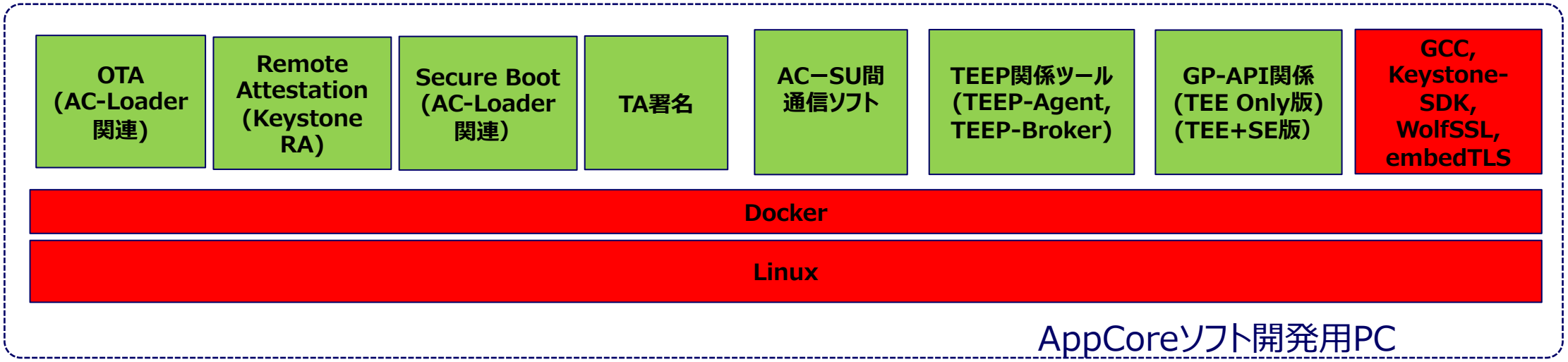
TRASIO研究技術の全体像

今後どうなる予定か（開発環境）





一般公開（既存オープンソース）
一般公開（オープンソース、設計、検証）
一般公開（バイナリ配布）
コミュニティ公開（ソースコード）
コミュニティ公開（API、外面）
TRASIO内部のみで共有
各社社外秘



TRASIO研究技術の全体像

まとめ

- TRASIOでは安全なIOTサービスを構築するために必要な
  - ハードウェア
  - 基盤ソフトウェア
  - 応用ソフトウェアを作る仕組み
  - 応用ソフトウェアを管理する仕組み
  - 鍵管理の仕組み
- と
  - それらを使って実現できるサービス
- をRISC-V CPUを使って実装するための研究をしています。