IPA Information-technology Promotion Agency, Japan

CC評価のセキュリティアーキテクチャ

2015年7月28日

独立行政法人情報処理推進機構
技術本部　セキュリティセンター
情報セキュリティ認証室

本講座の目的

① セキュリティアーキテクチャの概念の解説
② 「セキュリティアーキテクチャ記述」の解説

- CC（Common Criteria）とは
  - ITセキュリティ評価の国際標準規格（ISO/IEC 15408）
  - 多くの国でIT製品の調達要件として採用
- CC評価の内容
  - セキュリティ機能要件が正確・完全に実装されていること
  - セキュリティ機能が改ざん・バイパスされないこと
  - その他（製品マニュアル、開発・製造環境、製品の配送、etc）

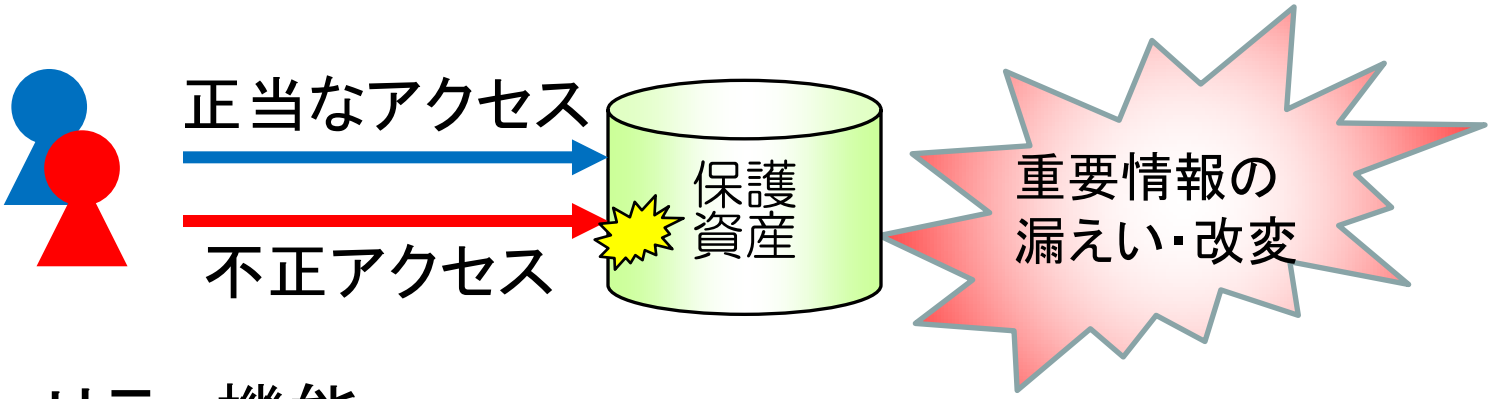
目次

- セキュリティアーキテクチャ
- セキュリティアーキテクチャ記述
  - バイパス防止
  - セキュリティドメイン
  - 改ざん防止（自己保護）
  - セキュアな初期化
- CC評価の脆弱性評定
- おわりに

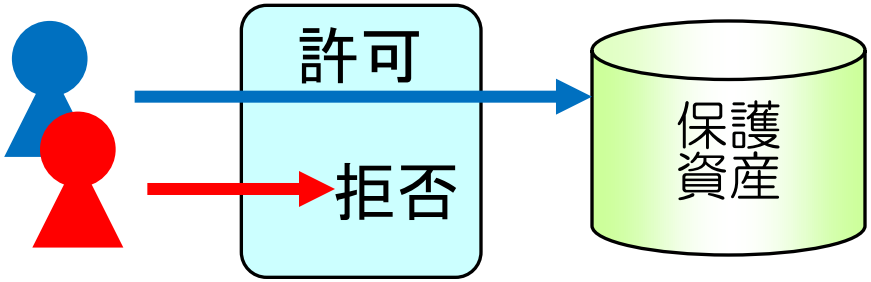
セキュリティアーキテクチャ

セキュリティ機能

脅威



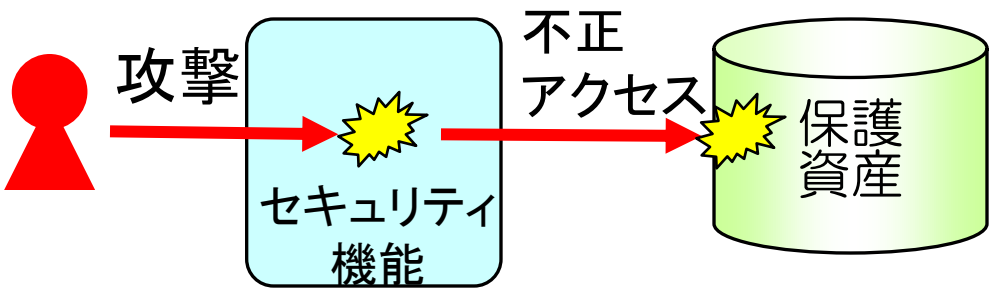
セキュリティ機能



セキュリティ機能
（識別認証、アクセス制御など）

セキュリティ機能だけでは不十分

セキュリティ機能の改ざん

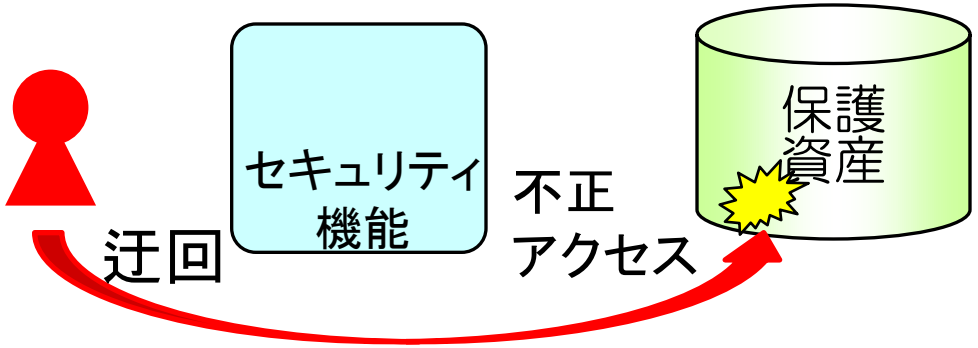


CC評価の「**改ざん**」は「tampering」の訳

不正な変更だけでなく、不正な干渉全般を含む。

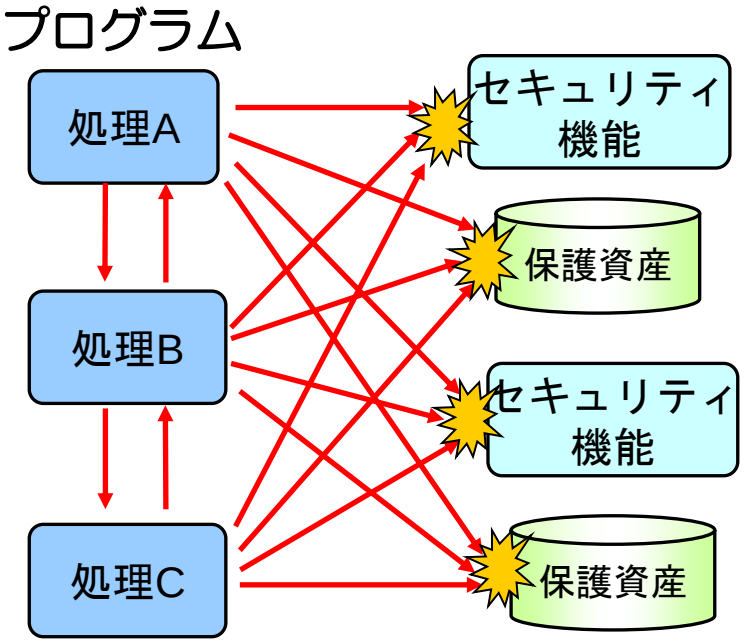
・バッファオーバフロー等の攻撃
・スクリプト等の注入による意図しないプログラム実行
・etc

セキュリティ機能のバイパス



セキュリティ機能を守るためには

プログラムの様々な欠陥を想定し、セキュリティ機能に
干渉する可能性のあるすべての経路の対策が必要

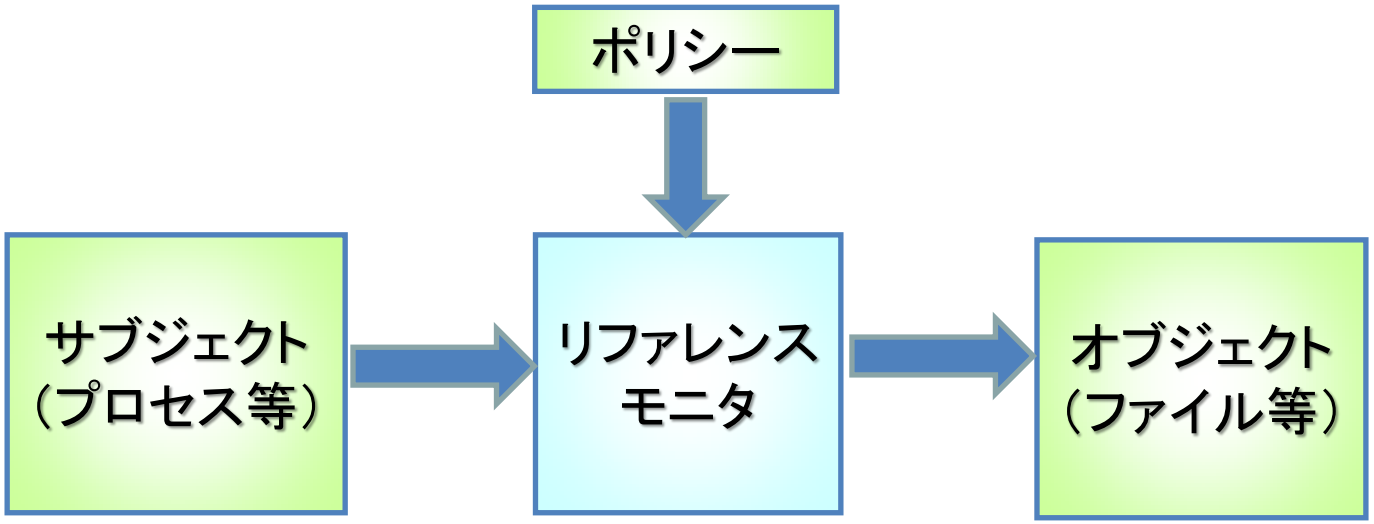


セキュリティアーキテクチャとは

- セキュリティ機能を安全に実装するための
  設計方針、構造、しくみ
- セキュリティ機能を攻撃から守る
  - 改ざん防止
  - バイパス防止
- 「リファレンスモニタ」の実現

J.P. Anderson, Computer security technology planning study, ESD-TR-73-51, 1972

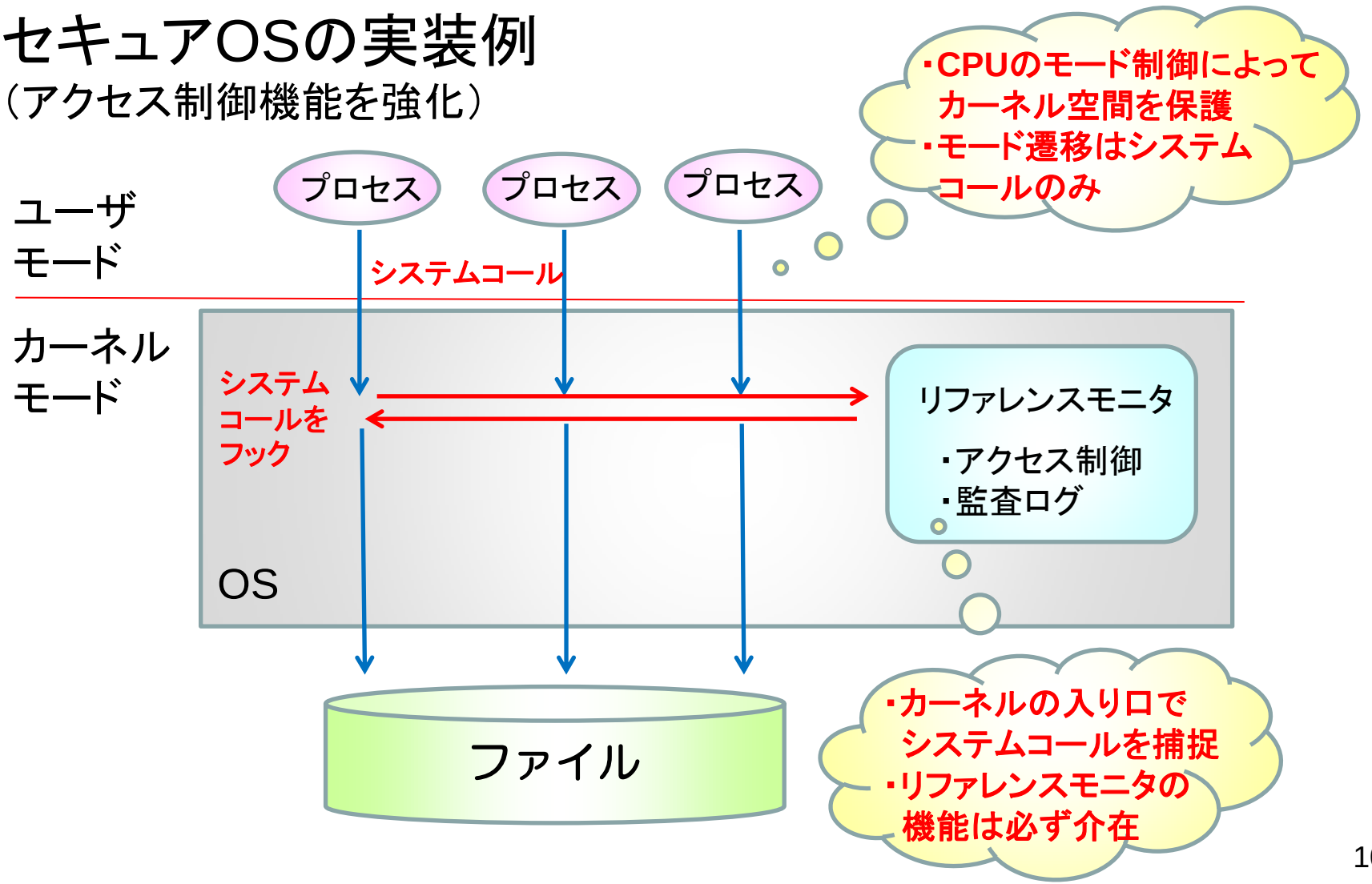
リファレンスモニタ



- リファレンスモニタの要件
  - ✓ Always invoked: 必ず介在する（バイパス防止）
  - ✓ Tamperproof: 改ざん防止
  - ✓ Verifiable: 検証可能
    （コンパクトに実装され保証可能）

リファレンスモニタの例

セキュアOSの実装例
（アクセス制御機能を強化）

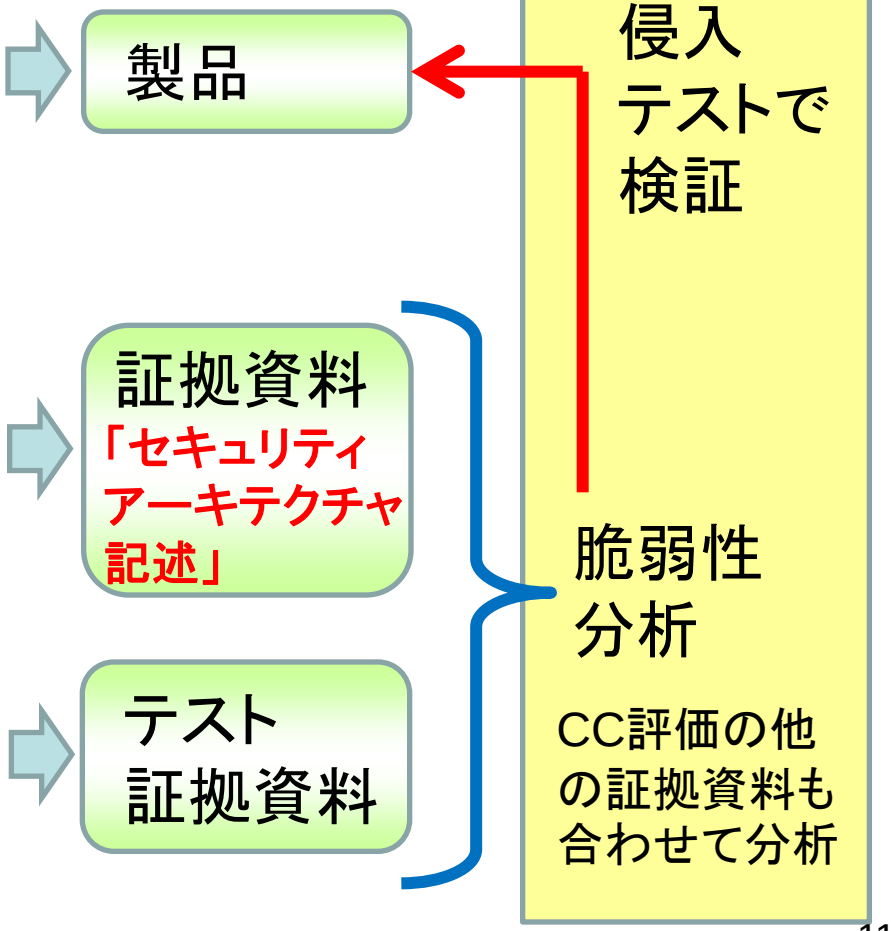


CCの要求内容

EAL: 評価保証レベル
レベルが高いほど詳細に検査される

開発者

- ✓ セキュリティ機能が改ざん・バイパスされないように設計実装すること
- ✓ 上記の設計実装の内容を決められた観点で記述し提供すること (EAL2以上の場合)
- ✓ 上記の設計実装の内容をテストし、証拠資料を提供すること (EAL3以上の場合)



（参考）CC規格

TOE: 評価対象
TSF: 評価対象のセキュリティ機能

●EAL1

AVA_VAN.1.1D	開発者は、テストのためのTOE を提供しなければならない。
AVA_VAN.1.3E	評価者は、基本的な攻撃能力を持つ攻撃者からの攻撃にTOEが耐えられることを決定するために、識別された潜在的脆弱性に基づいて侵入テストを実施しなければならない。

●EAL2で追加

ADV_ARC.1.3D	開発者は、TSF のセキュリティアーキテクチャ記述を提供しなければならない。
ADV_ARC.1.1E	評価者は、提供された情報が、証拠の内容・提示に対するすべての要件を満たしていることを確認しなければならない。

●EAL3で追加

ATE_DPT.1.2C	テストの深さの分析は、TOE 設計内のすべてのTSF サブシステムがテストされていることを実証しなければならない。

適用上の注釈

注意

TSF のアーキテクチャ的な安全性の記述(セキュリティアーキテクチャ(ADV_ARC)での)で、特定のメカニズムが挙げられている場合、開発者が実行するテストは、そのメカニズムが実行され、記述どおりに動作することを示さなければならない。

セキュリティアーキテクチャ記述

バイパス防止

セキュリティドメイン

改ざん防止（自己保護）

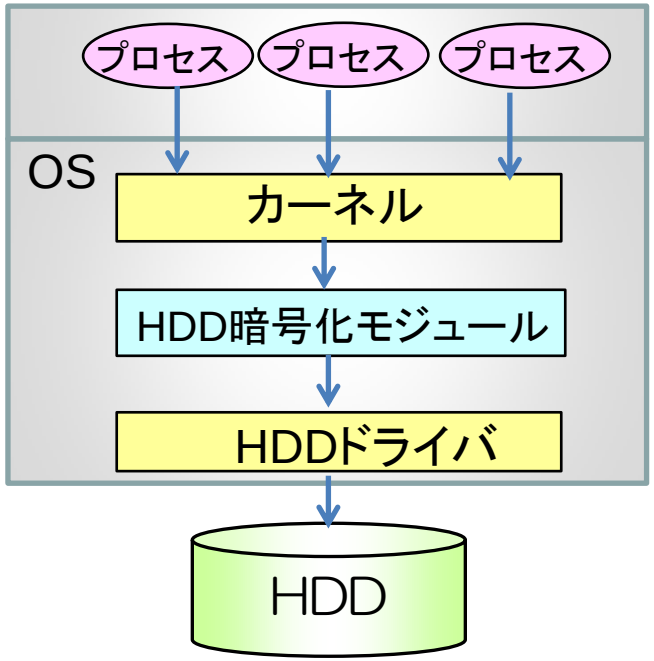
セキュアな初期化

バイパス防止

- セキュリティ機能がバイパスできないしくみ
- 製品のすべてのインタフェースとセキュリティ機能について、論証が必要
  - 保護資産にアクセス可能なインタフェースは、セキュリティ機能が必ず介在するしくみを説明すること
  - 保護資産やセキュリティ機能に関係のないインタフェースは、処理内容を示し、なぜ関係ないのかを説明すること

バイパス防止：例（1）

HDD暗号化
の実装例



前提

ユーザモードのプロセスが自由にアクセスできる空間は限定されている。

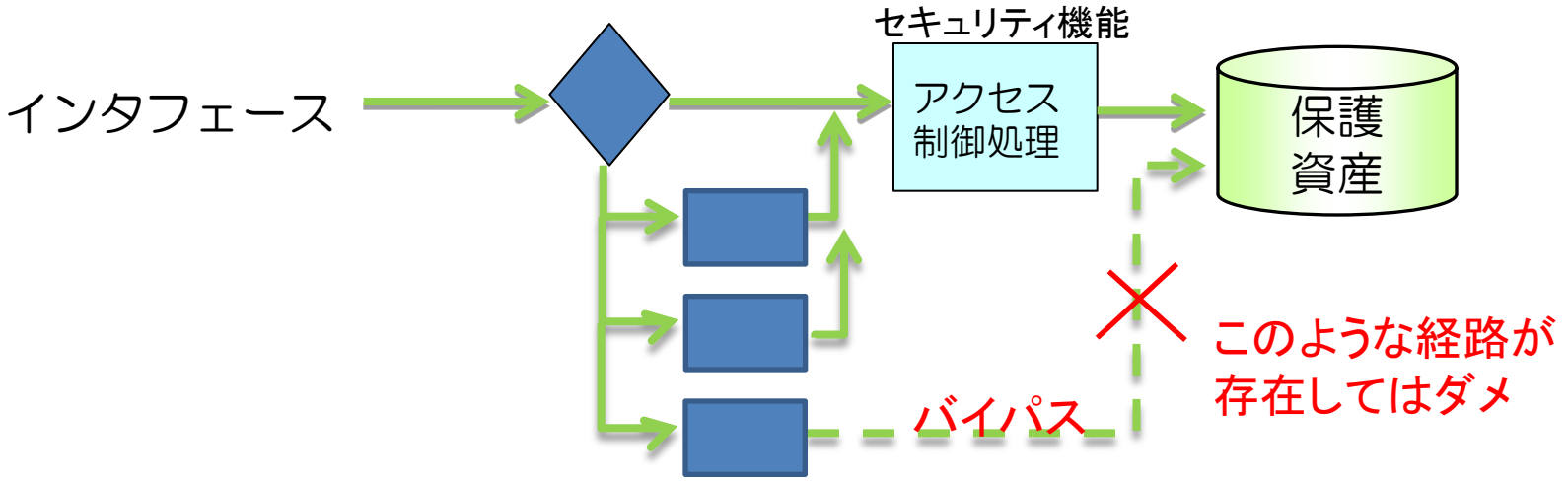
（セキュリティドメイン）

記述例　注：本例は理解を助けるための例示。証拠資料には具体的なしくみを記述。

ファイルやHDDの読み書きは、すべて、カーネルからHDDドライバを経由する構造になっている。HDDやHDDドライバに直接アクセスするインタフェースはない。HDD暗号化モジュールは、その構造を利用し、カーネルとHDDドライバの間に挿入されている。そのため、プロセスがファイルやHDDに読み書きする際に、HDD暗号化モジュールは必ず介在する。

バイパス防止：例（2）

保護資産にアクセス可能なインタフェース



セキュリティ機能が適用されずに保護資産にたどり着く経路が存在しないことを、具体的な処理内容を示して説明する。

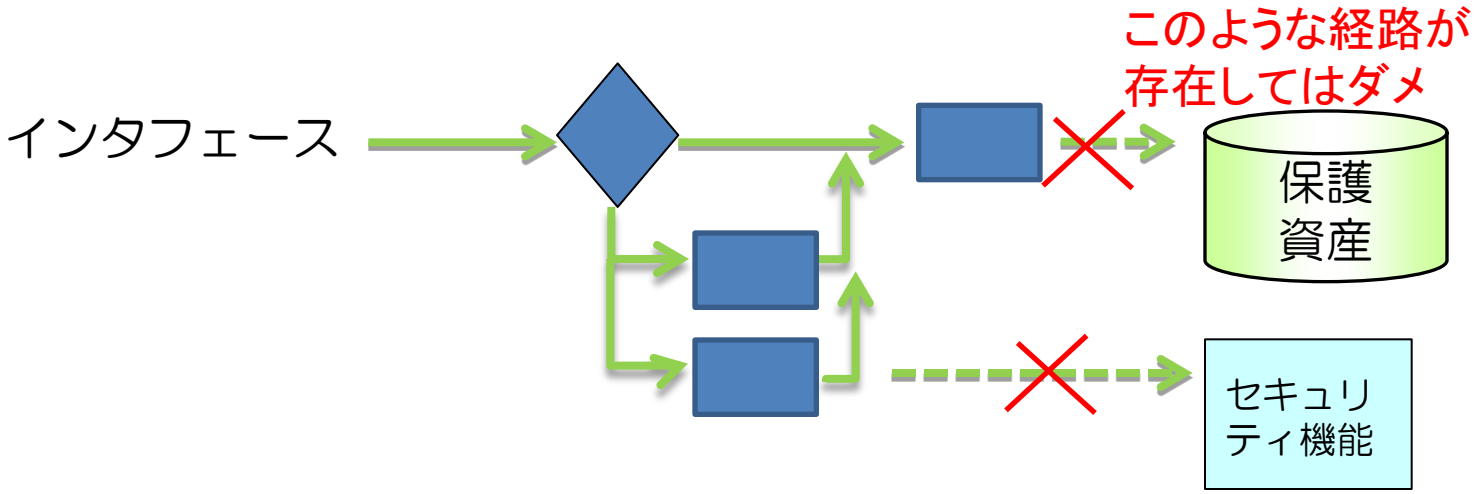
記述例　注：本例は理解を助けるための例示。証拠資料には具体的なしくみを記述。

処理は逐次的に行われ、処理途中で分岐はないため、セキュリティ機能は必ず適用される。

処理中に、入力パラメタをチェックし、入力パラメタの指定に応じて、アクセスするデータの種類や範囲を変更する処理がある。その処理では、入力パラメタから情報を取り出しているだけであり、保護資産へのアクセスは含まない。

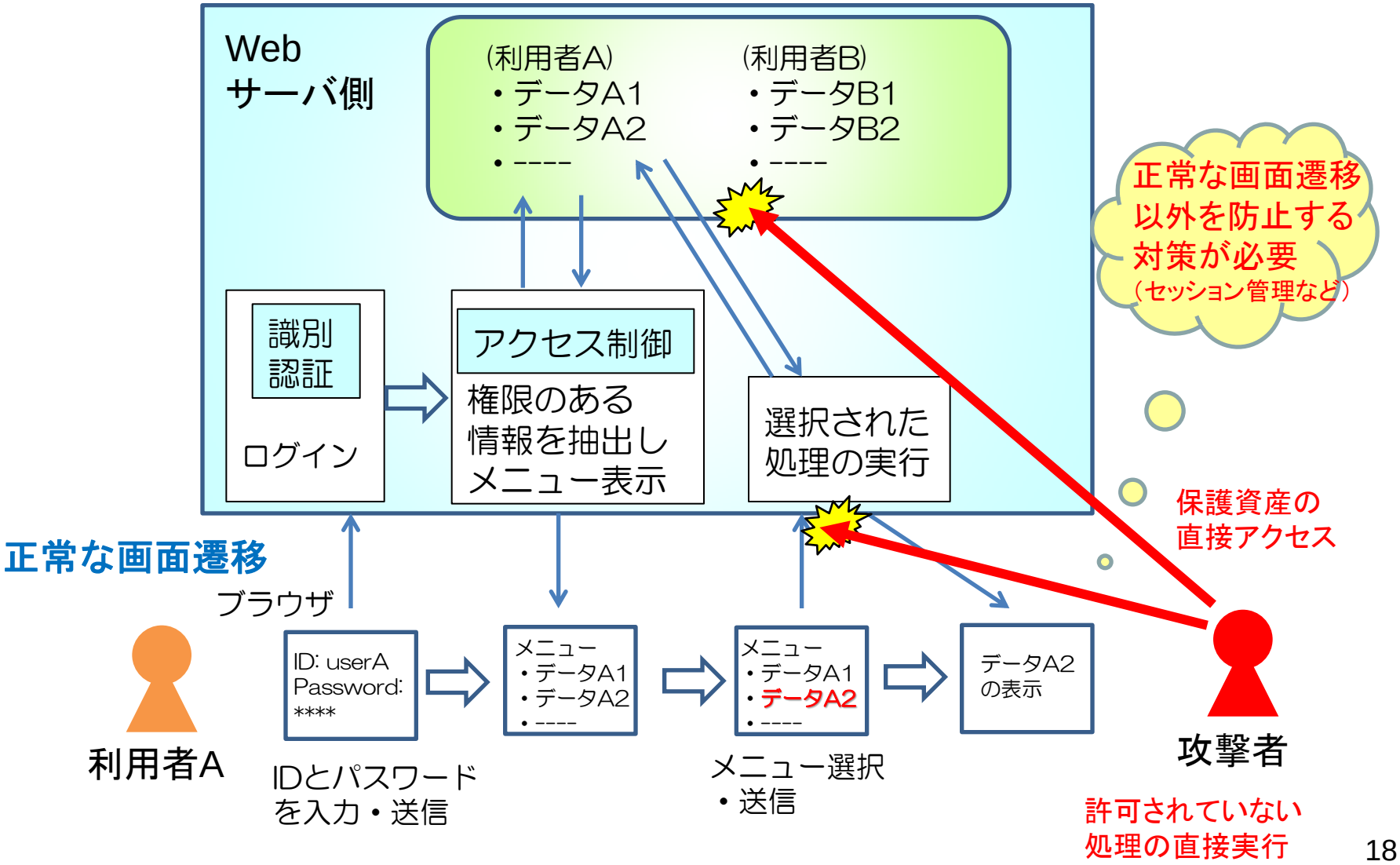
バイパス防止：例（3）

保護資産などとは関係のないインタフェース

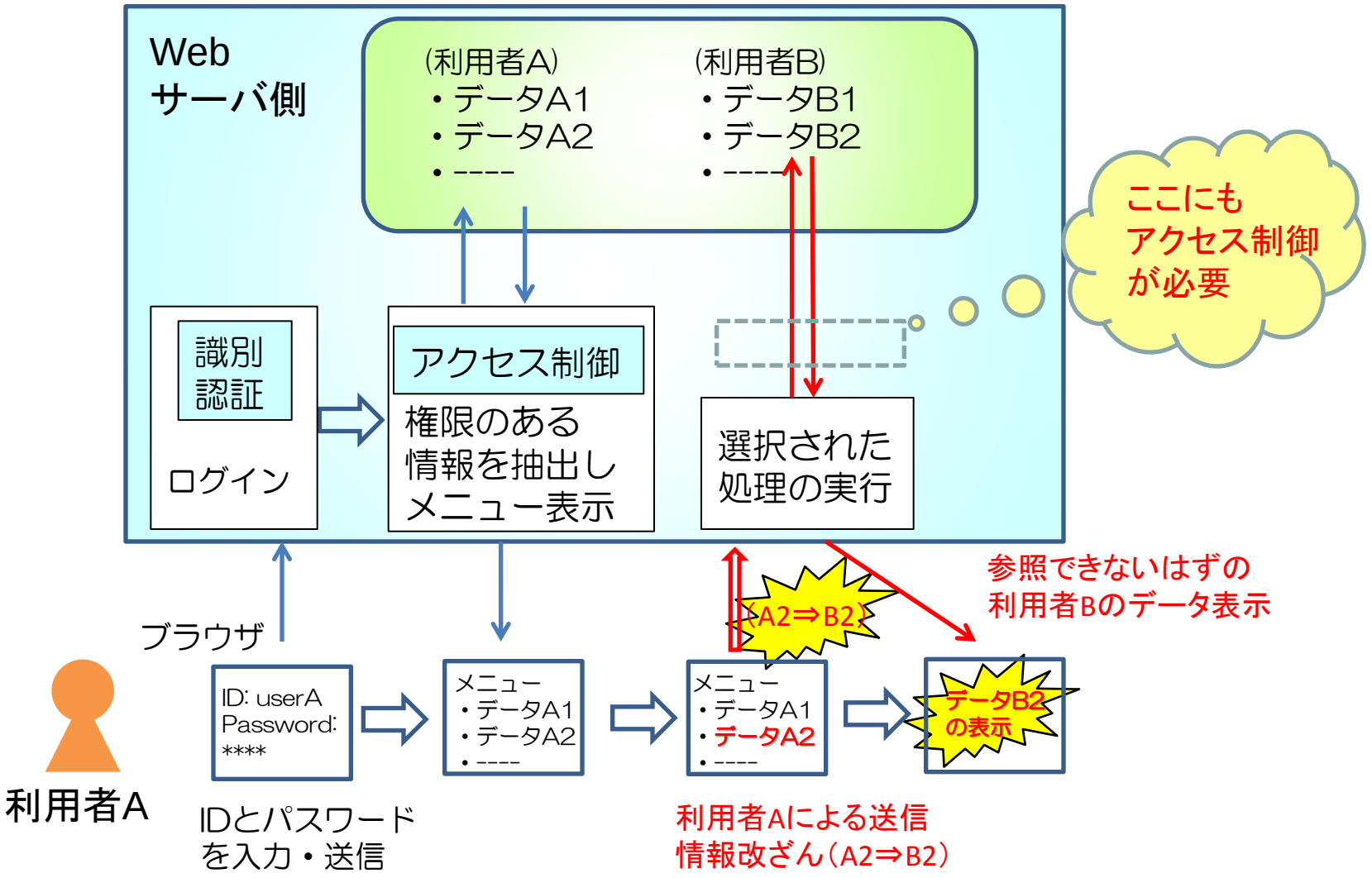


保護資産やセキュリティ機能にたどり着く経路が存在しないことを、
具体的な処理内容を示して説明する。

Webのバイパス：例（1）


Web
サーバ側
(利用者A)
・データA1
・データA2
・----
(利用者B)
・データB1
・データB2
・----
識別
認証
ログイン
アクセス制御
権限のある
情報を抽出し
メニュー表示
選択された
処理の実行
正常な画面遷移
以外を防止する
対策が必要
（セッション管理など）
保護資産の
直接アクセス
正常な画面遷移
ブラウザ
ID: userA
Password:
****
メニュー
・データA1
・データA2
・----
メニュー
・データA1
・データA2
・----
データA2
の表示
利用者A
IDとパスワード
を入力・送信
メニュー選択
・送信
攻撃者
許可されていない
処理の直接実行

Webのバイパス：例（2）


Web
サーバ側
(利用者A)
・データA1
・データA2
・----
(利用者B)
・データB1
・データB2
・----
ここにも
アクセス制御
が必要
識別
認証
ログイン
アクセス制御
権限のある
情報を抽出し
メニュー表示
選択された
処理の実行
ブラウザ
(A2⇒B2)
参照できないはずの
利用者Bのデータ表示
ID: userA
Password:
****
メニュー
・データA1
・データA2
・----
メニュー
・データA1
・データA2
・----
データB2
の表示
利用者A
IDとパスワード
を入力・送信
利用者Aによる送信
情報改ざん（A2⇒B2）

Webのバイパス：対策

- 対策内容
  - 保護資産への直接アクセスを防止する
  - ログインしていない人のアクセスを防止する（セッション管理）
  - 利用者入力は信用せずに必ず検証する
- 公知の脆弱性に注意が必要
  - ディレクトリトラバーサル
  - セッション管理
    - セッション情報の推測、セッションフィクセーション
    - クロスサイトリクエストフォージェリ等
  - TOCTOU (Time-of-check Time-of-use)

証拠資料の記述上の注意

- 具体的なしくみを記述すること

×

・セッション管理により、ログインした人以外はアクセスできない。
・セッション情報は乱数を使用しているため、推測できない。

○

・ログインが成功すると、ログインの通番、時間情報、ログインID、パスワードのSHA-256を計算し、セッションIDとして保持する。
・以降の画面遷移は、画面ごとに発行するワンタイムトークンを用いて、正当なアクセスを確認する。そのしくみは以下のとおり。
  - 画面ID、トークンのシリアル番号、セッションIDのSHA-256を計算し、トークンとして使用。
  - トークンの受け渡しには、HTMLのhiddenを使用。
  - 利用者から送られてきたトークンを、サーバ側で保持している値と比較し、一致した場合のみアクセスを許可する。

注：上記は証拠資料の詳細度を説明するための例示。例示したしくみの推奨ではない。

セキュリティアーキテクチャ記述

バイパス防止

セキュリティドメイン

改ざん防止（自己保護）

セキュアな初期化

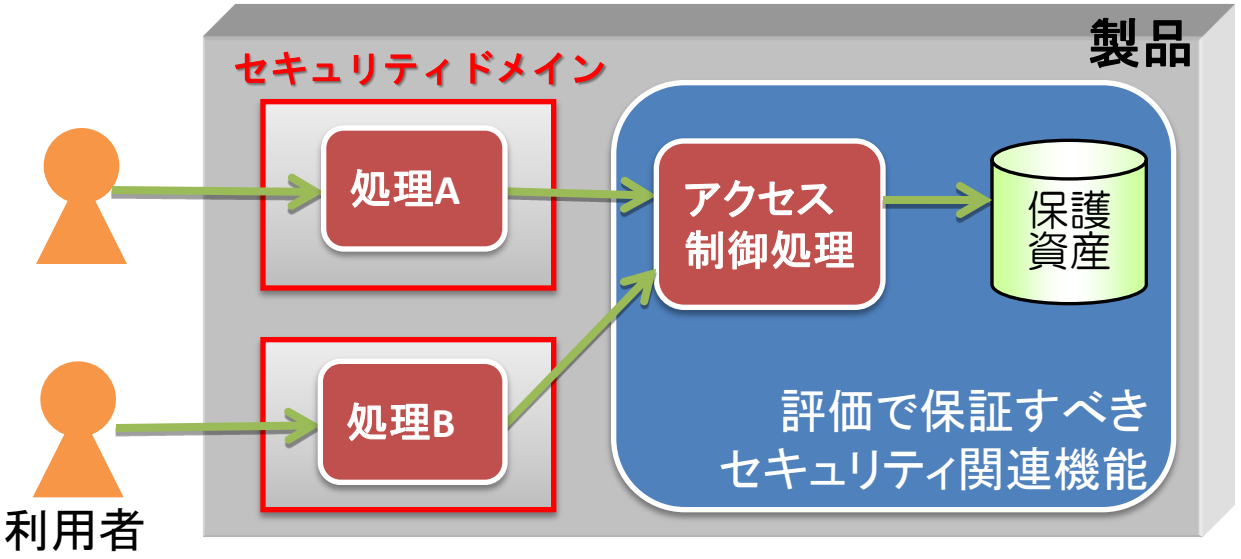
セキュリティドメイン（1）

- セキュリティ機能以外のプログラムを「閉じ込める」
  - 「セキュリティドメイン」とは、閉じ込められたプログラムが自由にアクセスできる環境や資源のこと
  - セキュリティ分野での「Sandbox」と同じ概念
- バイパス防止に役立つ
  - 「閉じ込める」ことにより、アクセス経路が限定される
  - 経路が限定されれば、セキュリティ機能が必ず介在する実装が容易（前述のセキュアOSやHDD暗号化の例）
- 改ざん防止に役立つ
  - 「閉じ込める」ことは、セキュリティ機能を守ること
  - 経路が限定されれば、保護すべき部分も局所化

セキュリティドメイン（2）

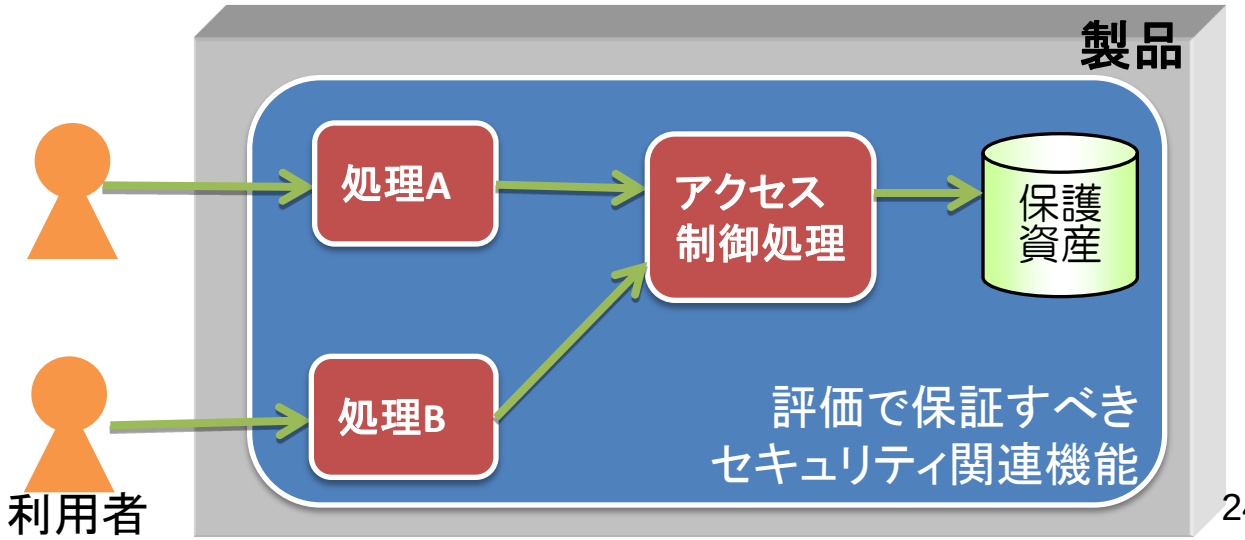
セキュリティドメインありの場合

・セキュリティ機能に関係ない処理を分離
・セキュリティ機能の安全性が高まる



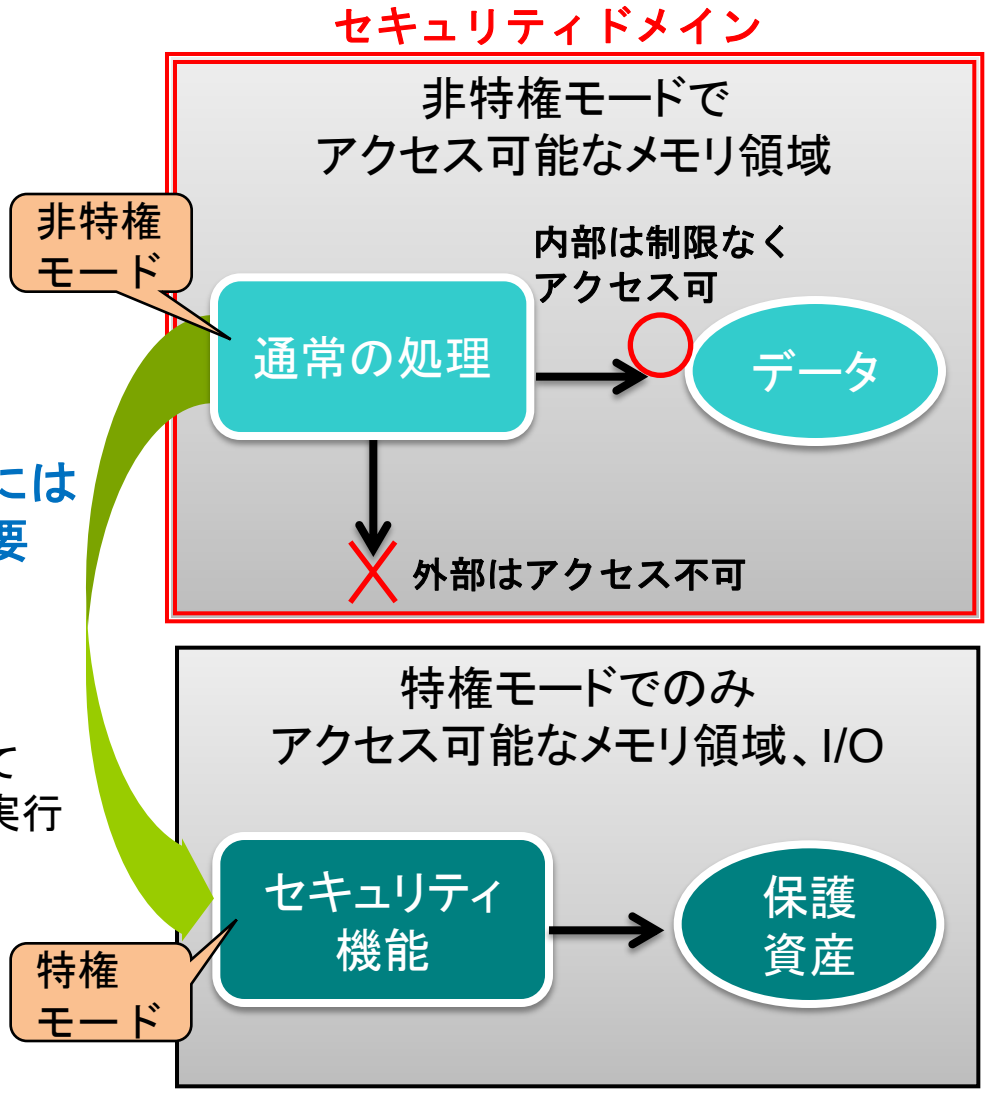
セキュリティドメインなしの場合

・一部分の処理の欠陥が、全体に悪影響
・安全性の保証のためには、セキュリティ機能と関係ない処理であっても、セキュリティ機能と合わせて検査が必要



セキュリティドメイン：例（1）

CPUの動作モード
による実現例

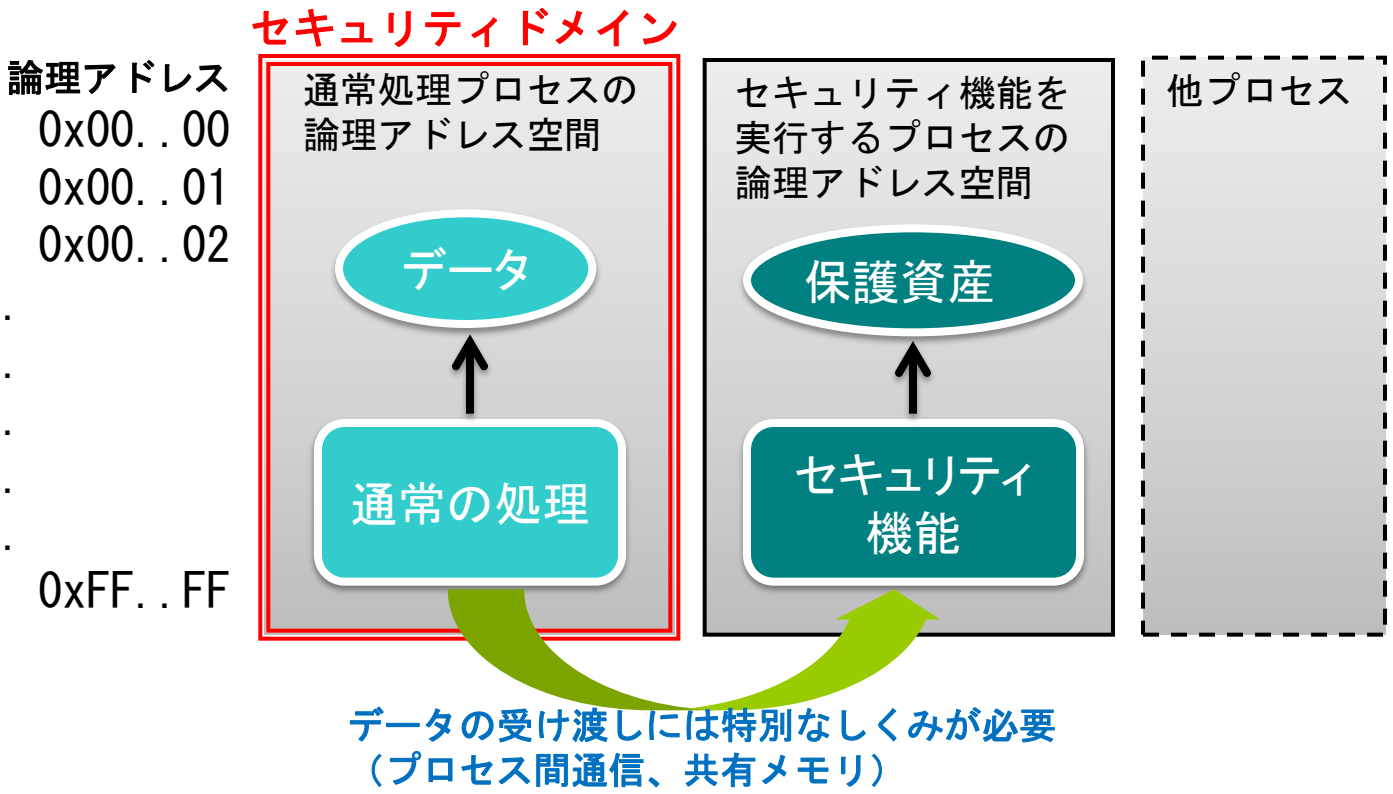


**データの受け渡しには
特別なしくみが必要
（システムコール）**

非特権モードから
特権モードに遷移して
セキュリティ機能を実行

セキュリティドメイン：例（2）

論理アドレス空間
による実現例

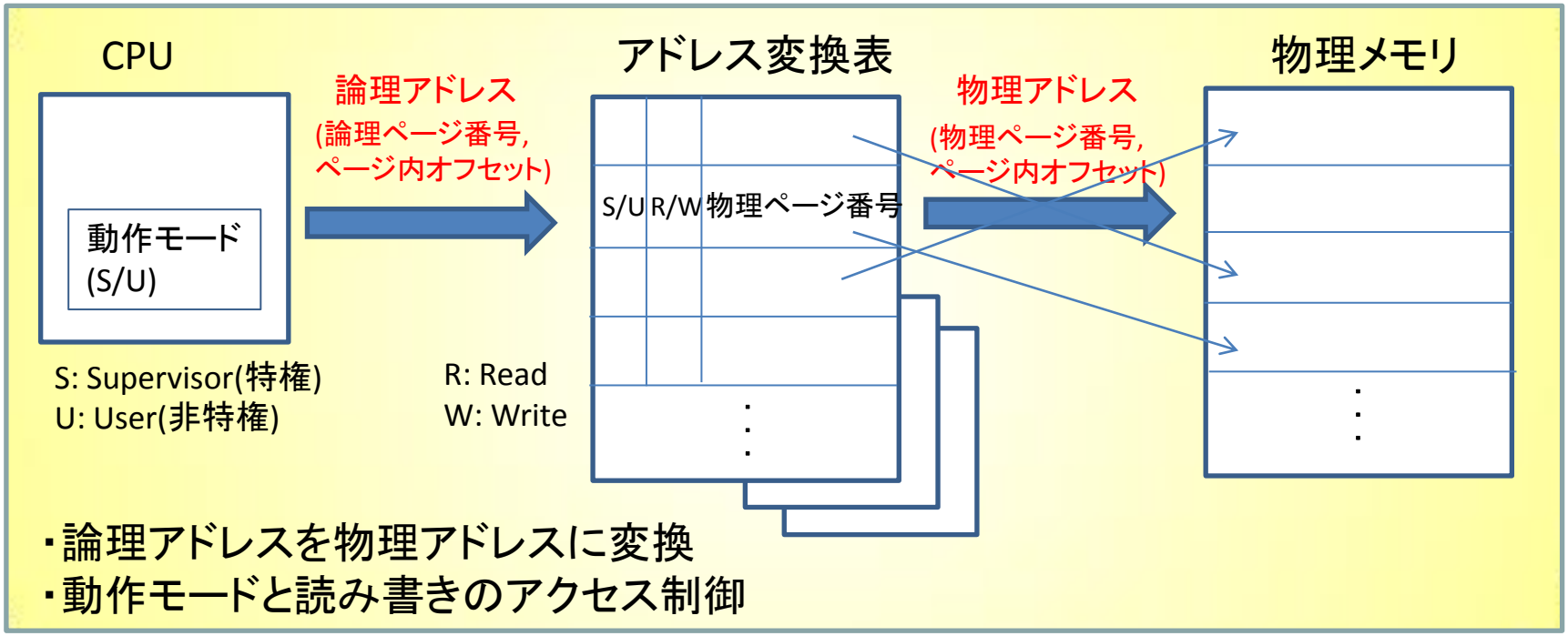


プロセス毎に論理アドレス空間が異なる
各プロセスのプログラムは自プロセスのアドレス空間だけが参照できる

例(1)と例(2)のメカニズム

IPA

ハードウェアのメカニズム



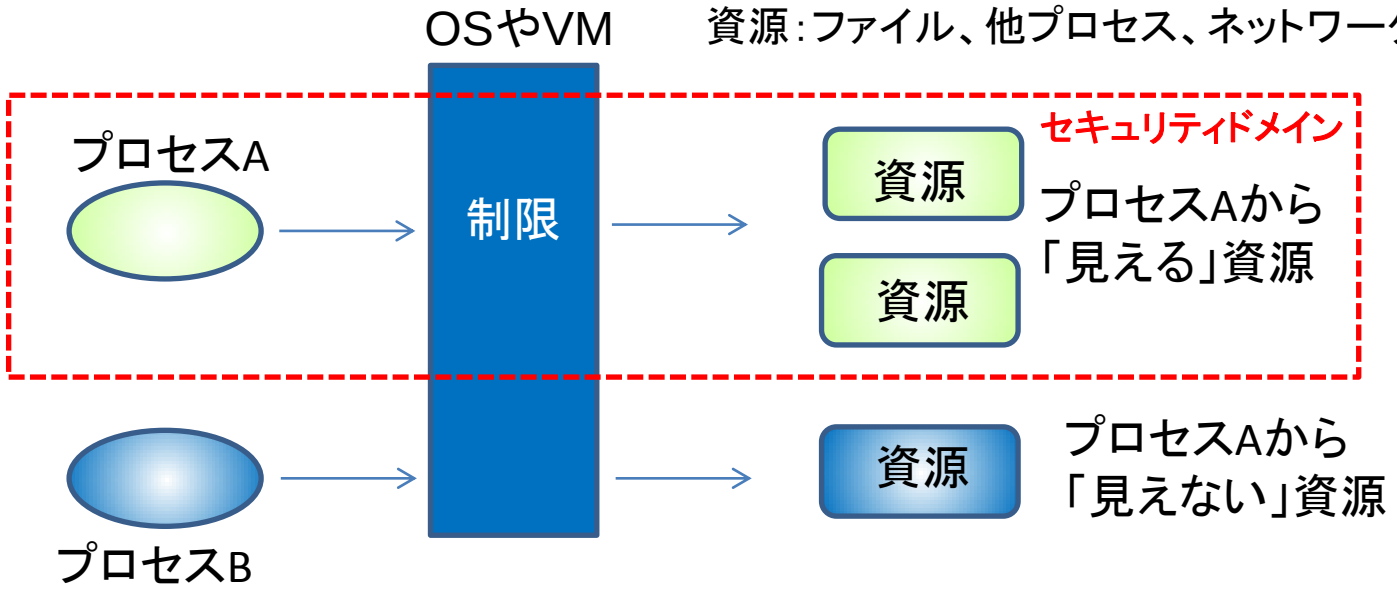
・論理アドレスを物理アドレスに変換
・動作モードと読み書きのアクセス制御

ハードウェアを制御するソフトウェア(OS等)

・プロセス毎にアドレス変換表を用意
・プロセスを切り替える際に、当該プロセス用のアドレス変換表を
ハードウェアに設定

セキュリティドメイン：例（3）

ソフトウェアによる実現例

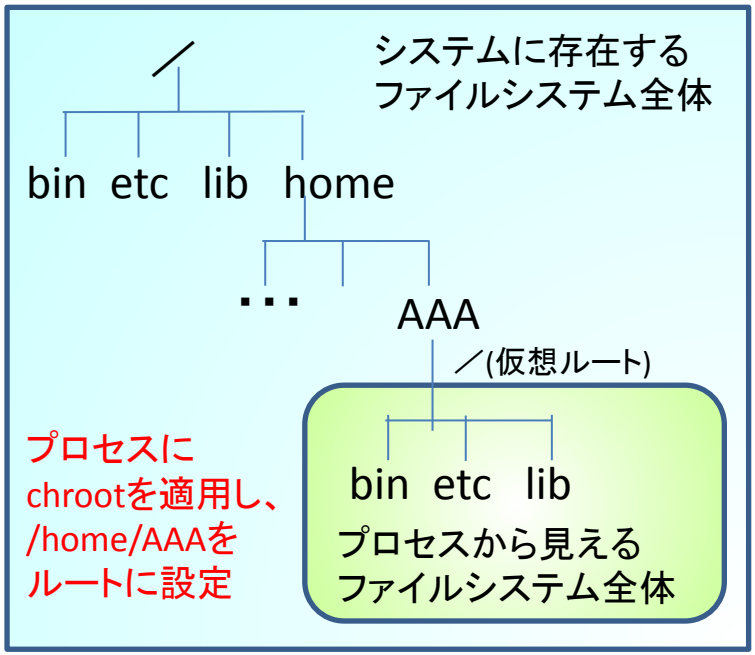


プロセスはOSやVMを介して
資源にアクセス

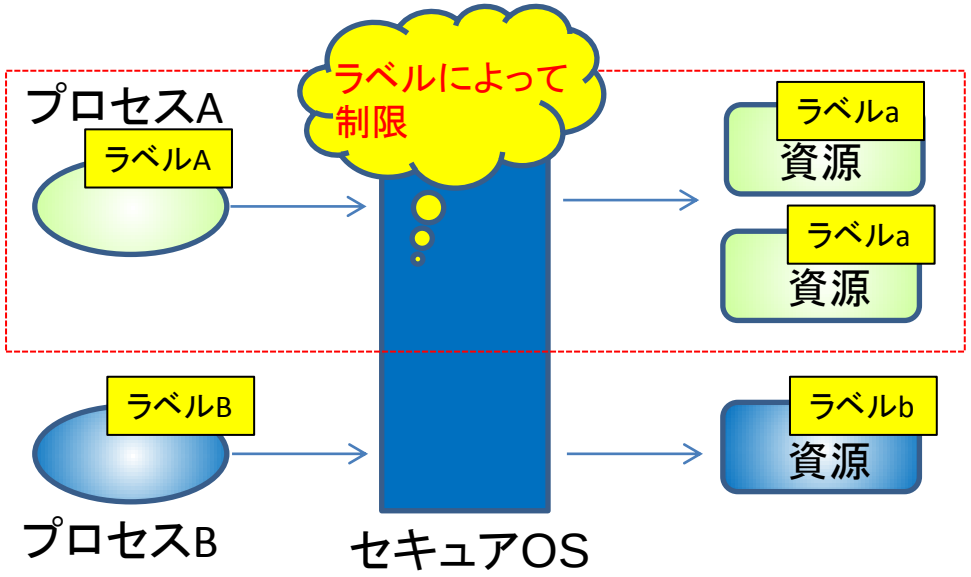
OSやVMによって制限すれば、
領域外の資源はアクセスできない
だけでなく、存在すらわからない

セキュリティドメイン：例（3）続き

単純な例（chroot）



汎用的な例（強制アクセス制御）



セキュリティドメイン：記述内容

- セキュリティドメインが存在する場合
  - セキュリティドメインの定義
    - セキュリティドメインの対象の処理
    - セキュリティドメインに含まれる資源
      - アドレス空間、ファイルなど
  - セキュリティドメインの分離メカニズム

- セキュリティドメインが存在しない場合
  - セキュリティドメインが不要である根拠
    - 入力が厳しく制約されていることが必要

セキュリティアーキテクチャ記述

バイパス防止

セキュリティドメイン

改ざん防止（自己保護）

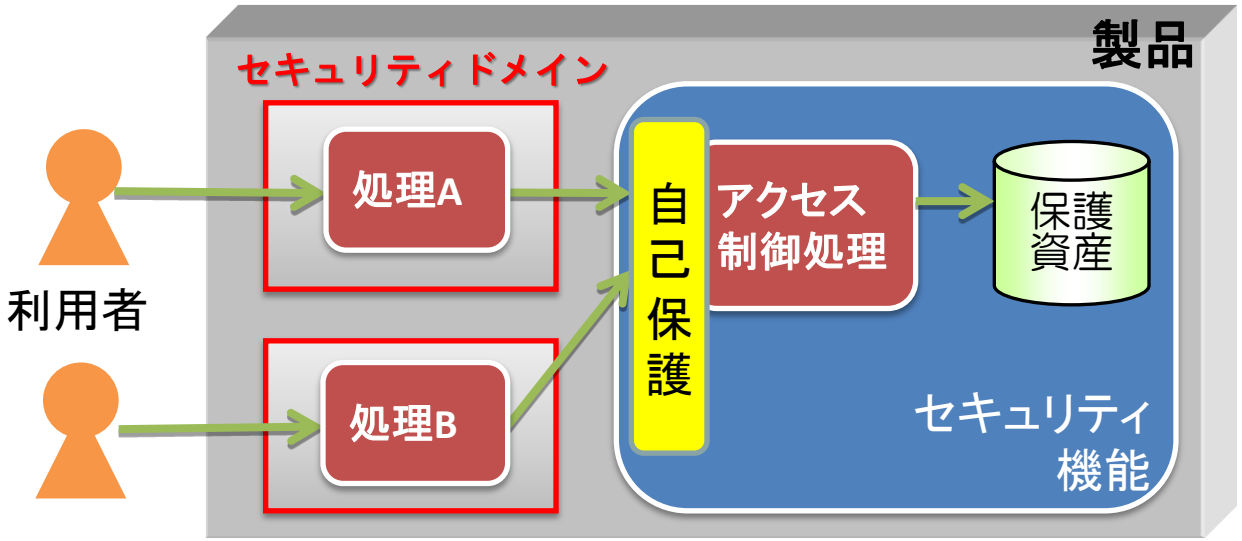
セキュアな初期化

改ざん防止（自己保護）

- セキュリティ機能を攻撃から守るすべてのしくみ
  - セキュリティドメインを分離するしくみ
  - セキュリティ機能以外のプログラムからの入力の内容をチェックしたり、安全に処理するしくみ
- ただし、製品自身が実現している部分
  - 外部環境に依存している場合には、製品側の分担している処理が該当
  - 製品と外部環境の依存関係を明確にすることが必要

改ざん防止（自己保護）（1）

製品単独でセキュリティドメインを実現している場合



改ざん防止（自己保護）

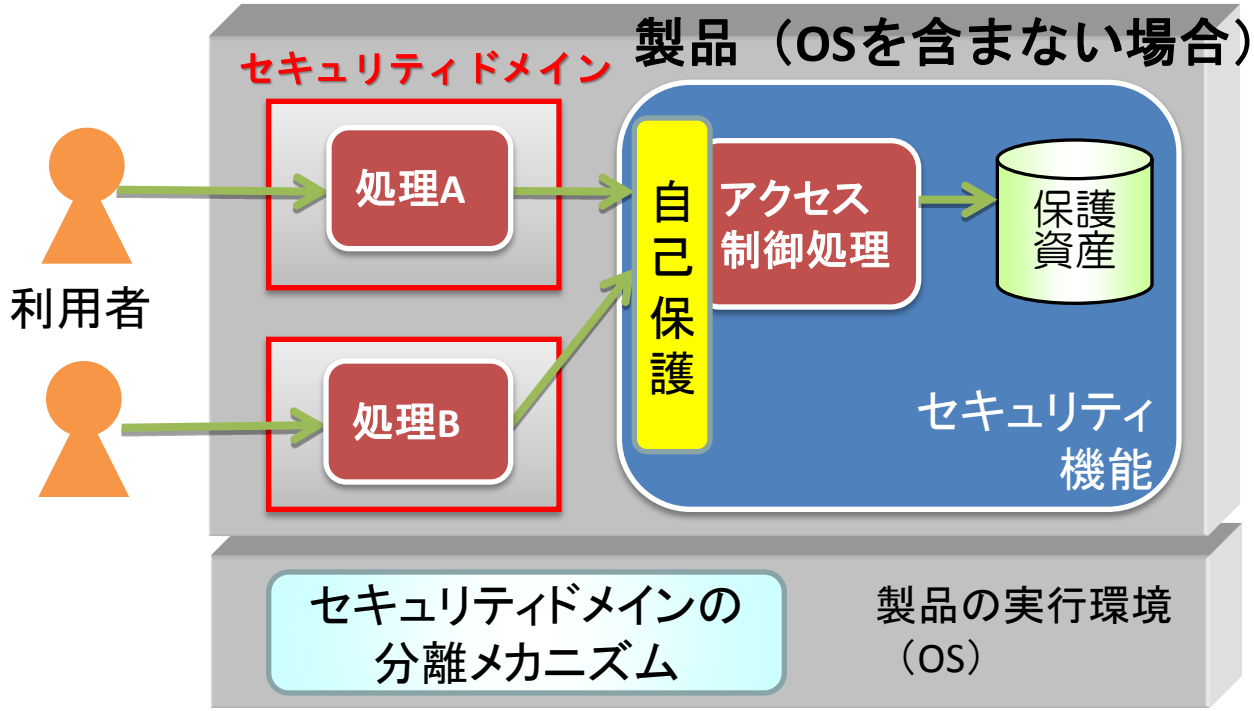
＝セキュリティドメインの分離メカニズム

＋セキュリティ機能への入力を保護するしくみ

改ざん防止（自己保護）（2）

製品が外部環境に依存している場合

「自己保護」は製品自身の実装部分



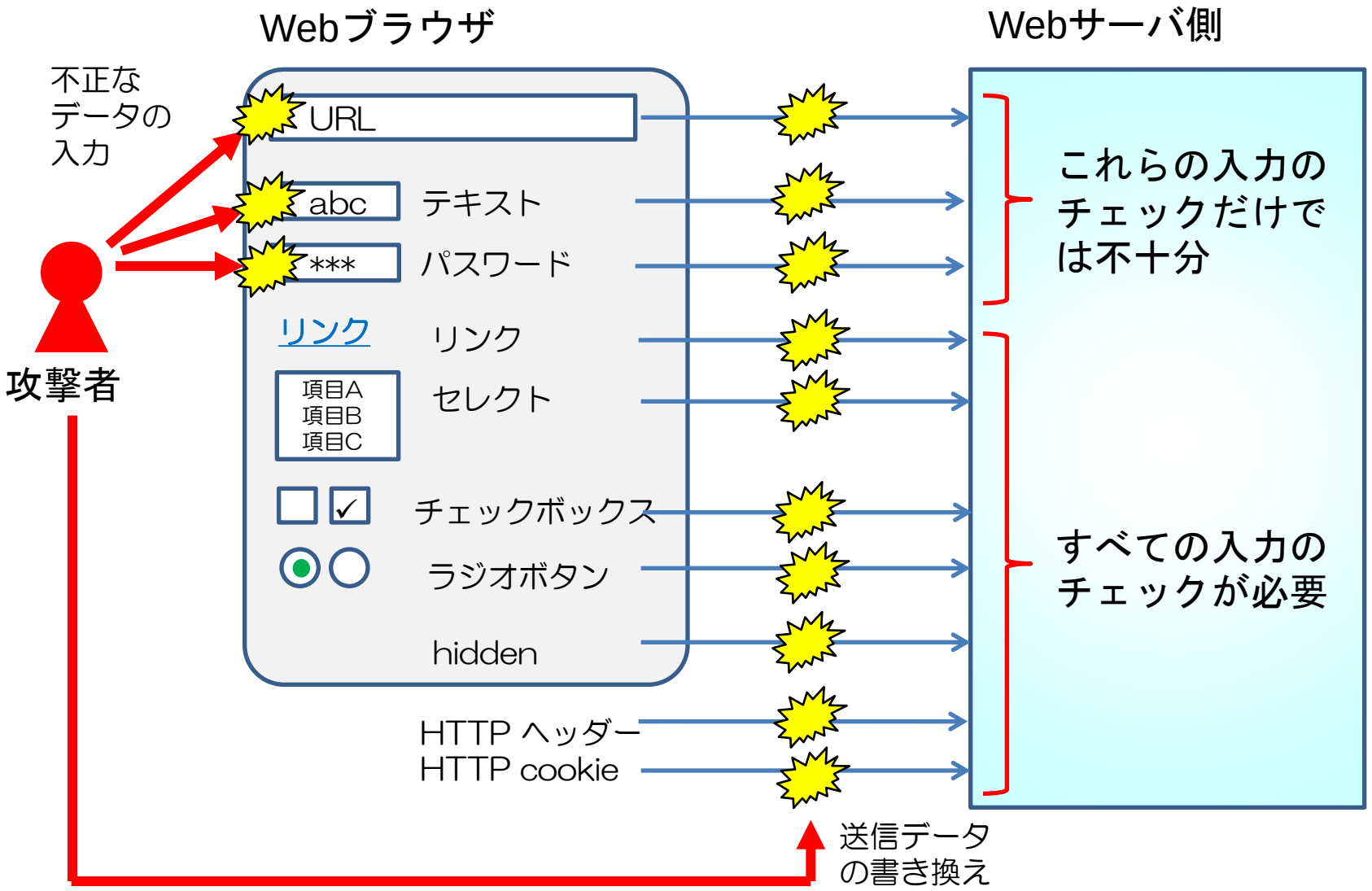
改ざん防止（自己保護）

＝セキュリティドメインを利用するしくみ

＋セキュリティ機能への入力を保護するしくみ

例えば、プロセスの生成、権限設定、処理の実行などが該当

Web入力の改ざん


Webブラウザ
Webサーバ側
不正な
データの
入力
攻撃者
URL
abc
テキスト
***
パスワード
リンク
リンク
項目A
項目B
項目C
セレクト
チェックボックス
ラジオボタン
hidden
HTTP ヘッダー
HTTP cookie
これらの入力の
チェックだけで
は不十分
すべての入力の
チェックが必要
送信データ
の書き換え

Web入力の改ざん：対策

- 対策内容
  - 利用者入力は信用せずに必ず検証する
- 公知の脆弱性に注意が必要
  - バッファオーバフロー
  - 各種インジェクション
    - SQL、OSコマンド、スクリプト、HTTPヘッダー、etc
  - クロスサイトスクリプティング（注：対策は出力時）
  - 各種エンコーディングの配慮

※証拠資料には具体的なしくみを記述する

セキュリティアーキテクチャ記述

バイパス防止

セキュリティドメイン

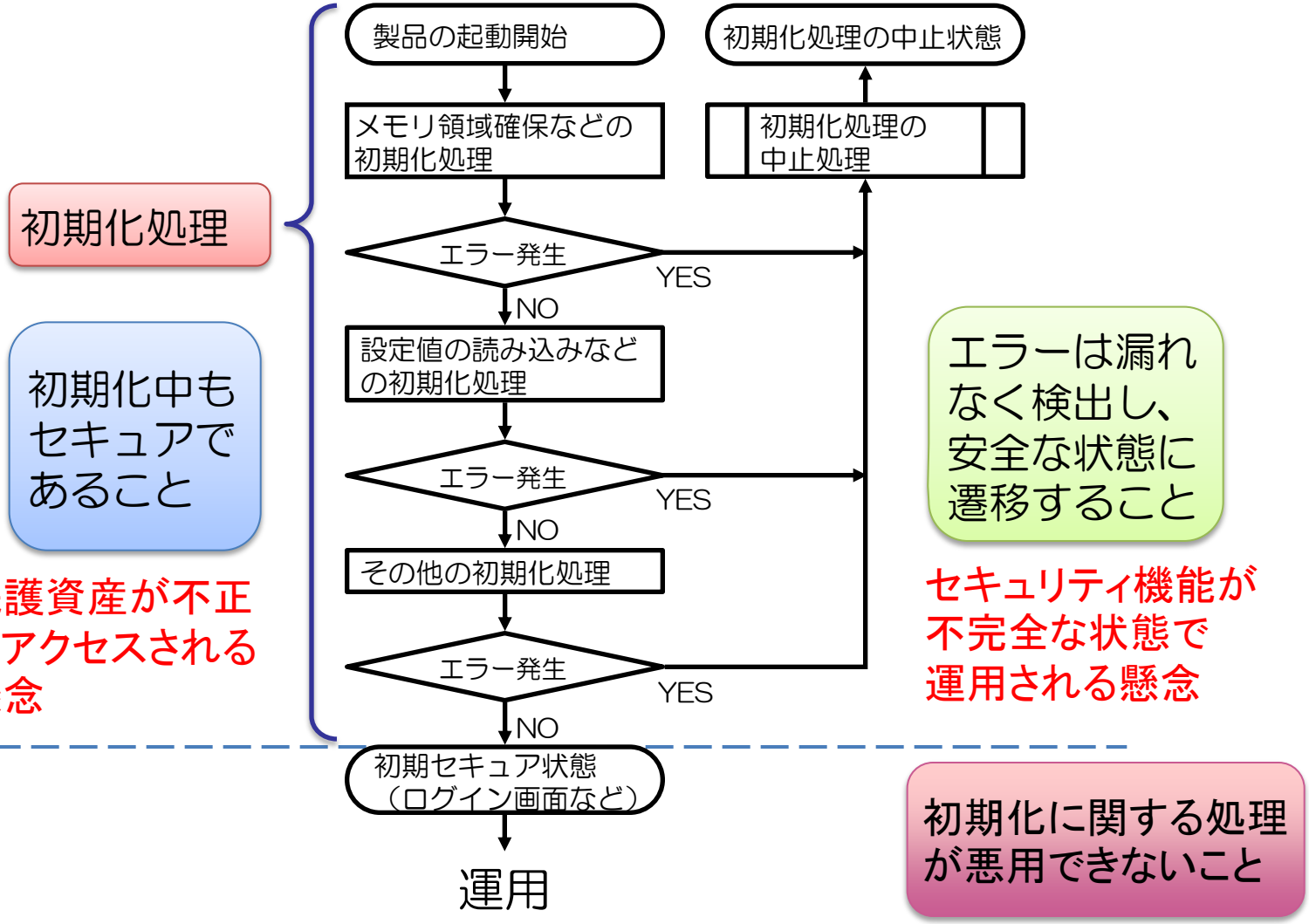
改ざん防止（自己保護）

セキュアな初期化

セキュアな初期化

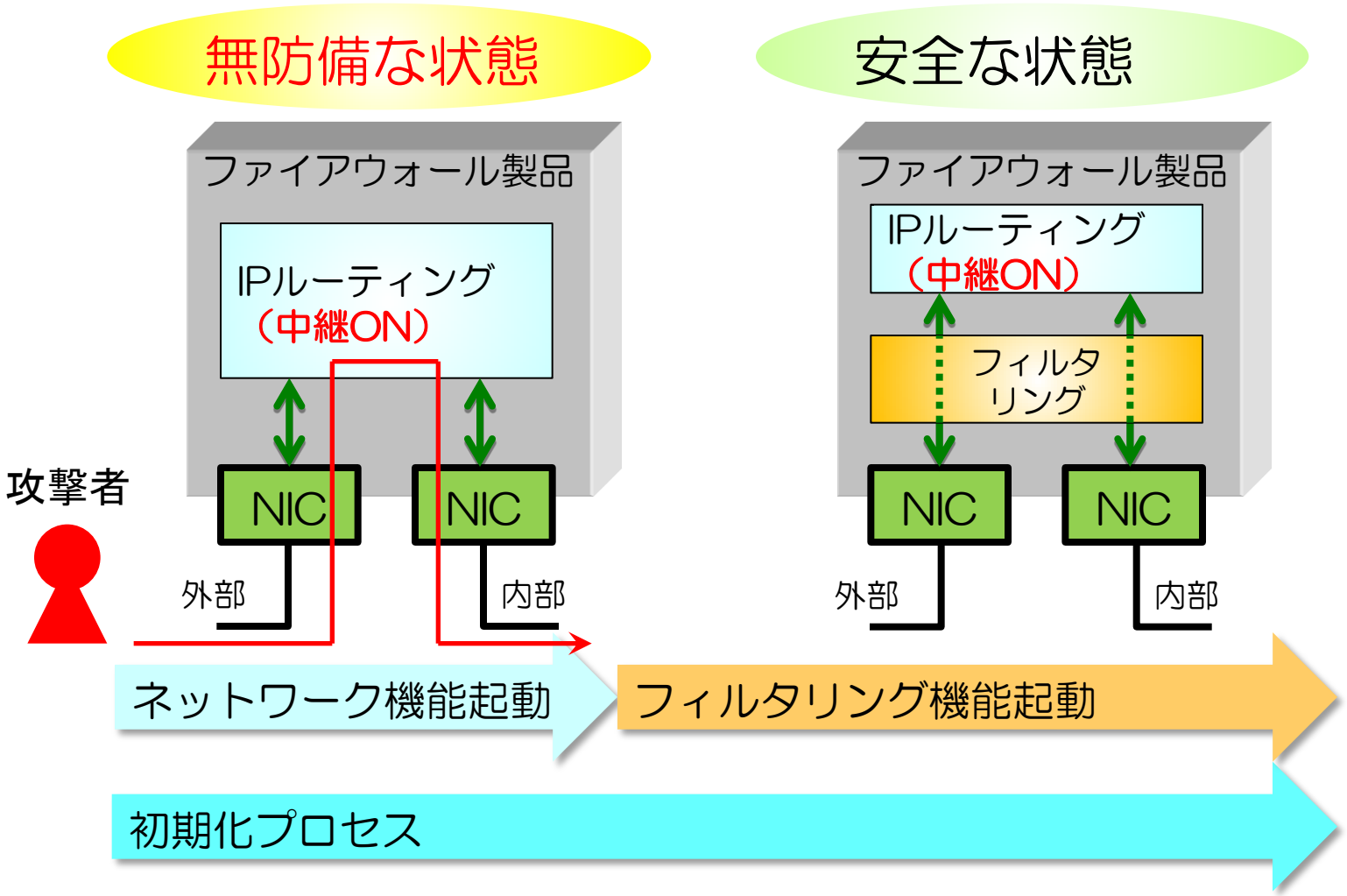
- 製品の起動途中のセキュリティ対策
  - セキュリティ機能はまだ動作していない
  - セキュリティの配慮を忘れがち
- 脅威
  - 起動途中の不正アクセス
  - 起動途中のエラー処理の不備
    （セキュリティ機能が不完全な状態で運用される懸念）
  - 初期化処理やデータの悪用

セキュアな初期化：観点


初期化処理
製品の起動開始
メモリ領域確保などの初期化処理
エラー発生
YES
NO
設定値の読み込みなどの初期化処理
エラー発生
YES
NO
その他の初期化処理
エラー発生
YES
NO
初期セキュア状態（ログイン画面など）
運用
初期化処理の中止状態
初期化処理の中止処理
初期化中もセキュアであること
保護資産が不正にアクセスされる懸念
エラーは漏れなく検出し、安全な状態に遷移すること
セキュリティ機能が不完全な状態で運用される懸念
初期化に関する処理が悪用できないこと

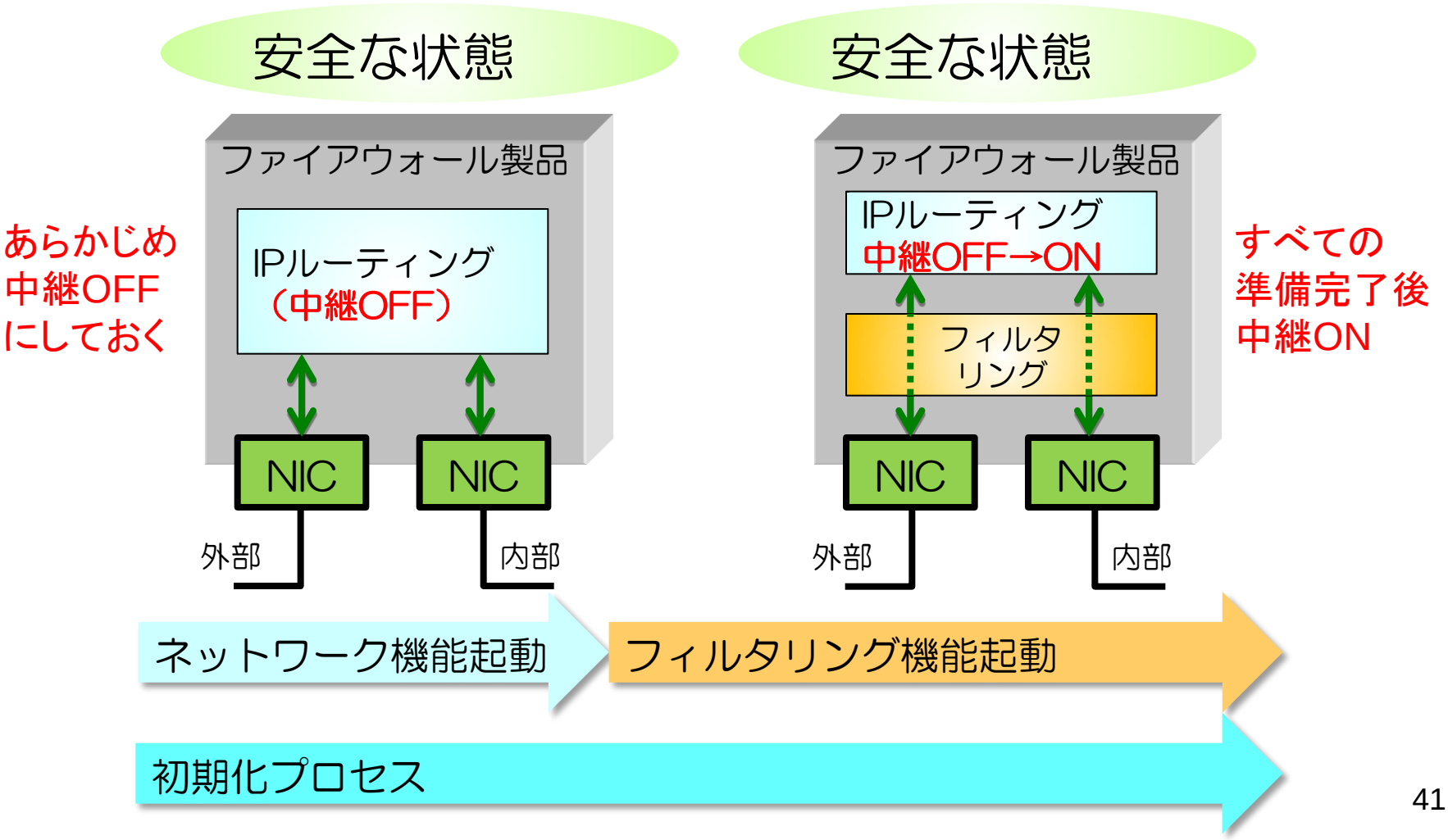
セキュアな初期化：例（1）

初期化中にセキュアでない状態に陥る例



セキュアな初期化：例（2）

初期化中もセキュアである例



セキュアな初期化：記述内容（1）

- 初期化処理の特定
  - 製品の起動方法（再起動や再開を含む）
  - 初期セキュア状態（例えば、ログイン画面の表示など）
  - 起動から初期セキュア状態までの処理概要
- セキュリティ機能の完全性を確保するしくみ
  - 攻撃やエラーが発生したとしても、不完全な状態にならないようにするための、具体的なしくみを記述
- 保護資産を保護するしくみ
  - 初期化中に保護資産にアクセスができないようにするための、具体的なしくみを記述

セキュアな初期化：記述内容（2）

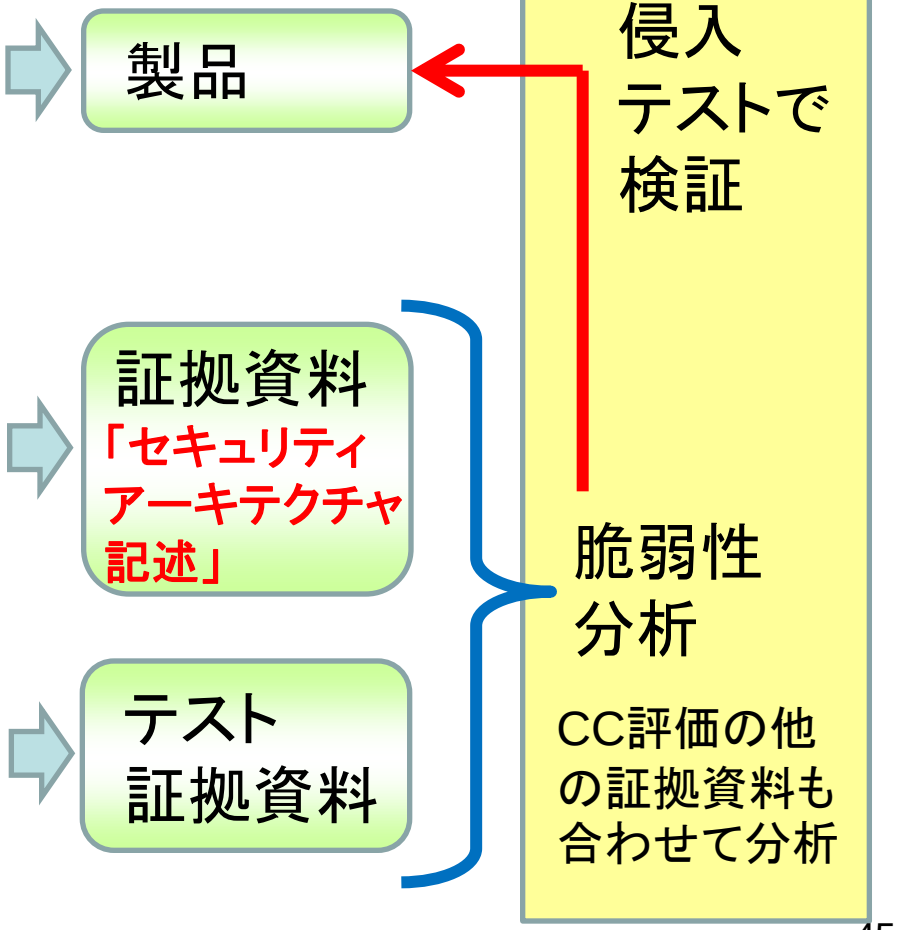
- 初期化処理の悪用を防止するしくみ
  - 製品起動後、初期化処理やそのデータにアクセスする手段を提供していない場合には、その旨を記述
  - 初期化処理やそのデータにアクセスする手段が存在する場合には、不正なアクセスを防止するしくみを記述
- 注意事項
  - 製品のしくみだけでは安全性を保証できない場合には、運用条件を記述する
  - 運用条件は、確実に実施されるように、製品のガイダンスで注意喚起することが必要

CC評価の脆弱性評定

CCの要求内容

開発者

- ✓ セキュリティ機能が改ざん・バイパスされないように設計実装すること
- ✓ 上記の設計実装の内容を決められた観点で記述し提供すること (EAL2以上の場合)
- ✓ 上記の設計実装の内容をテストし、証拠資料を提供すること (EAL3以上の場合)



脆弱性評定：合否の判定

攻撃の困難さを数値化した指標（攻撃能力）で合否を判定

所要時間

条件	評定値
～1日	0
～1週間	1
～2週間	2
～1ヶ月	4
～2ヶ月	7
～3ヶ月	10
～4ヶ月	13
～5ヶ月	15
～6ヶ月	17
6ヶ月～	19

専門知識

条件	評定値
しろうと	0
熟練者	3
エキスパート	6
複数のエキスパート	8

機会

条件	評定値
不必要/無制限	0
容易	1
中	4
困難	10
なし	∞

TOEの知識

条件	評定値
公開	0
限定的	3
機密	7
危機的	11

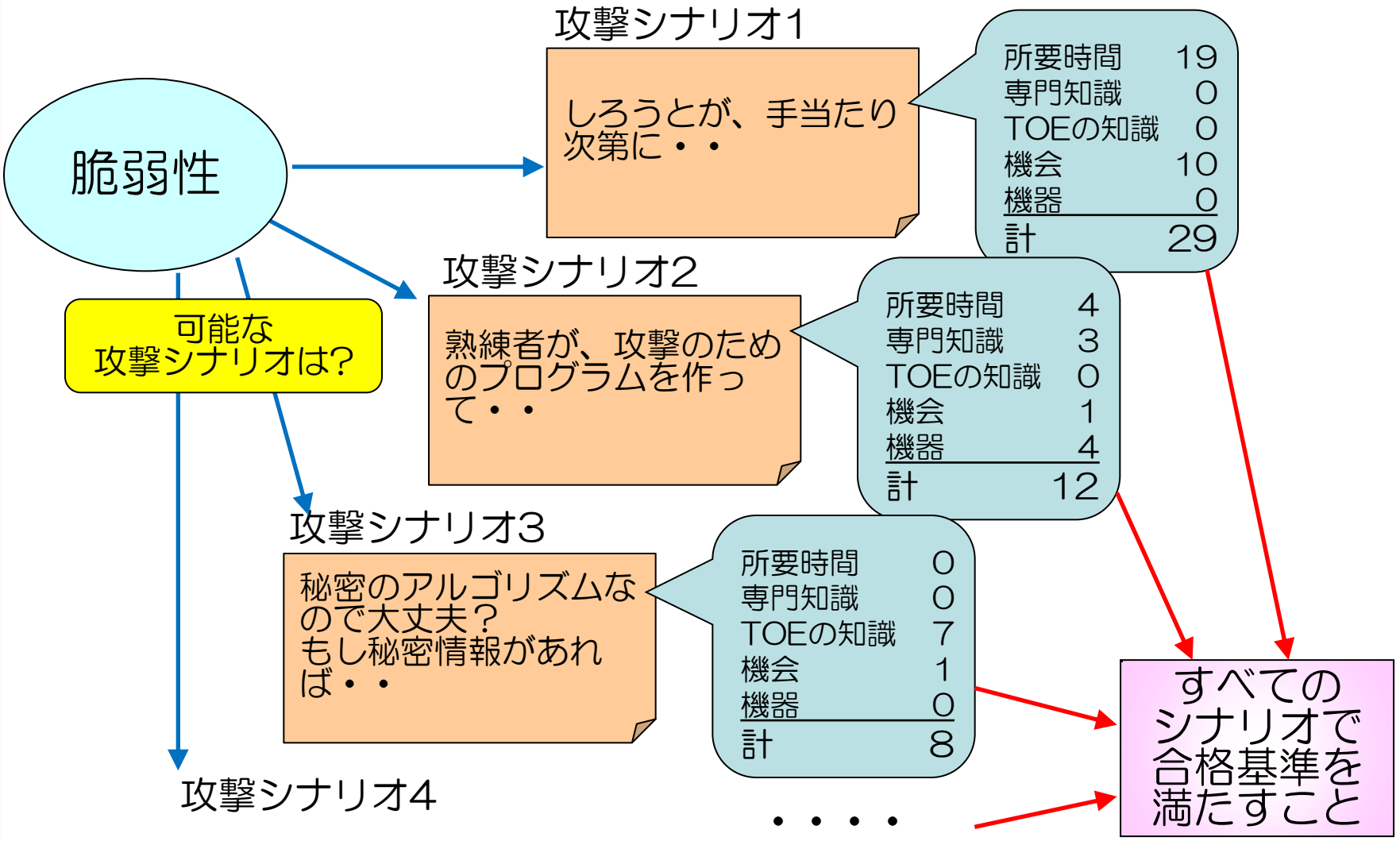
機器

条件	評定値
標準	0
特殊	4
特別注文	7
複数の特別注文	9

合計 → 攻撃能力

攻撃能力		
25～高より上		
20～24 高	EAL 1～5は合格	
14～19 中	EAL 1～4は合格	EAL 5以上は不合格
10～13 強化基本	EAL 1～3は合格	EAL 4以上は不合格
0～9 基本		全EALで不合格

脆弱性評定：攻撃能力の計算例



おわりに

まとめ

- セキュリティアーキテクチャ
  - セキュリティ機能の改ざんやバイパスを防止する
  - 適切なアーキテクチャにより、製品の安全性が高まり、その保証が容易になる
- セキュリティアーキテクチャ記述
  - 具体的なしくみを記述
    - セキュリティ機能と同等の詳細度で、設計・実装内容を記述することが求められている
    - 設計資料（CC評価の他の証拠資料）を参照する形式でも良い
  - 公知の脆弱性の配慮が必要

参考資料

- CC/CEM規格

  http://www.ipa.go.jp/security/jisec/cc/index.html

- 開発者のためのセキュリティ解説書

  http://www.ipa.go.jp/security/jisec/apdx.html#SEC_GUIDE

  – セキュリティアーキテクチャ編
  – 脆弱性評価編

- 上記解説書で紹介している各種参考資料

ご清聴ありがとうございました

お問い合わせ
E-mail: jisec@ipa.go.jp