HSM

# IBM Power + karta HSM

Przykłady użycia — IBM 4770 w trybie CCA



# Cel i mapa spotkania

## 1 · Fundament

- Czym HSM różni się od akceleracji CPU
- Stos Power → CCA → 4770
- Granica bezpieczeństwa

## 2 · Sześć scenariuszy

- PKI / podpis
- Klucze aplikacyjne
- Szyfrowanie i RNG
- TR-31 / EMV
- HA master keys
- PQC

## 3 · Decyzja

- Kiedy warto
- Kiedy nie warto
- Wymagania wdrożeniowe
- Rekomendowany pilot

# HSM, TPM i akceleracja CPU — trzy różne role

## HSM / IBM 4770

- Klucze operacyjne i master keys
- Operacje wykonywane wewnątrz urządzenia
- Kontrola dostępu i separacja obowiązków
- Odporność na manipulację

## TPM / vTPM

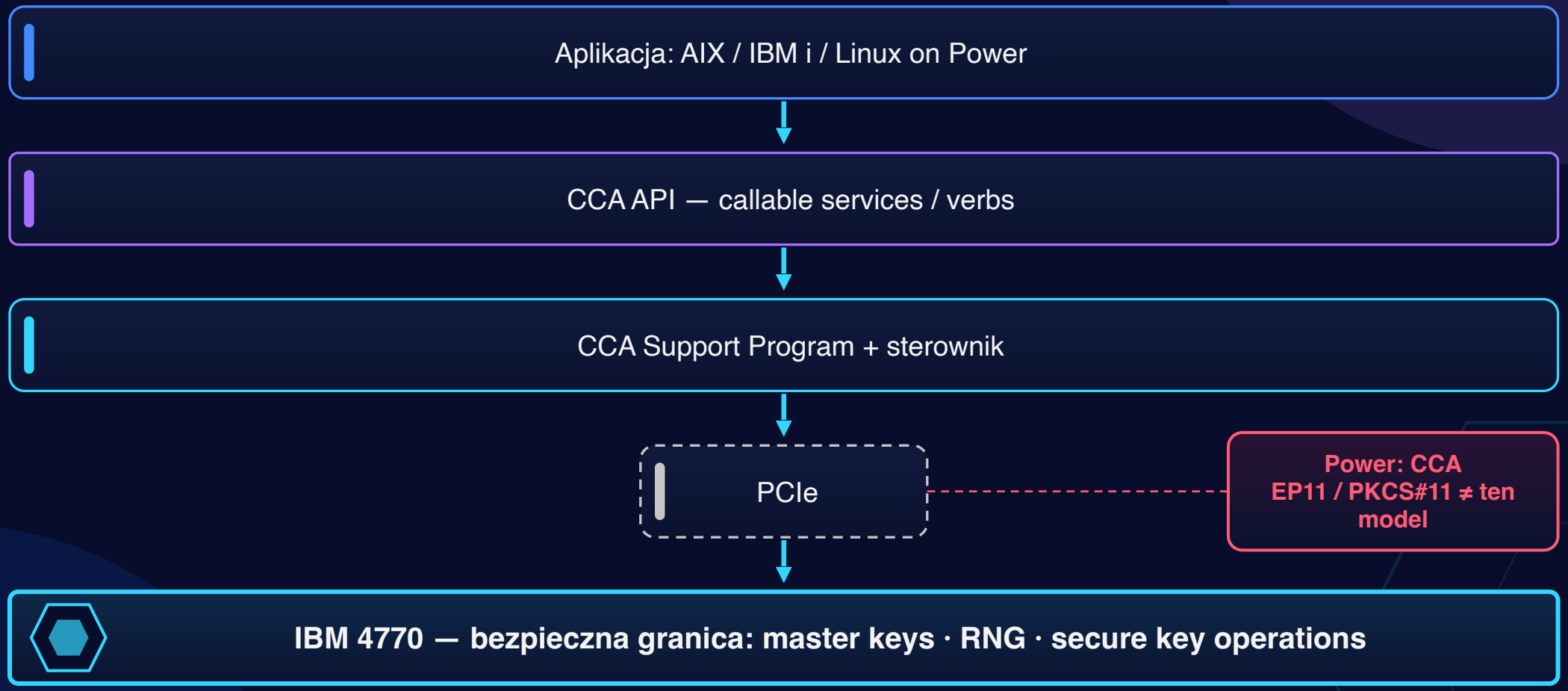
- Pomiary i atestacja platformy
- Secure / Trusted Boot
- Tożsamość urządzenia lub LPAR
- Nie zastępuje HSM aplikacyjnego

## CPU / akceleratory

- Wydajność kryptografii masowej
- Szyfrowanie pamięci i danych
- Klucze zwykle w domenie OS/aplikacji
- Brak dedykowanej polityki HSM

**Wybór 4770 uzasadnia potrzeba ochrony cyklu życia klucza — nie sam fakt używania kryptografii.**

# Architektura na IBM Power: aplikacja → CCA → 4770




**Na IBM Power karta 4770 działa w trybie CCA — EP11 / PKCS#11 nie jest tu właściwą ścieżką.**

# Model kluczy CCA: co faktycznie chroni HSM

## Master keys

- Niezależne rodziny m.in. dla kluczy symetrycznych, asymetrycznych i AES
- Rejestry: current / old / new
- Służą do ochrony kluczy roboczych

## Secure key token

- Materiał klucza jest opakowany pod master key
- Aplikacja przechowuje token, nie jawny klucz
- Operacja kryptograficzna trafia do HSM

## Kontrola użycia

- Typ i przeznaczenie klucza
- Access Control Points / role
- Return code + reason code
- Audyt i procedury dual control

**Najważniejszy wzorzec: aplikacja przechowuje token; HSM utrzymuje kontrolę nad użyciem klucza.**

# Sześć scenariuszy

01

## PKI i podpis

Tożsamość i niezaprzeczalność

02

## Klucze aplikacyjne

Oddzielenie klucza od systemu

03

## Szyfrowanie + RNG

Klucze danych i jakość losowości

04

## TR-31 / EMV

Kontrolowany transport kluczy

05

## HA master keys

Ciągłość operacji po awarii

06

## PQC

Migracja algorytmów

**Wybrane przypadki obejmują wartość bezpieczeństwa, ciągłość działania i przyszłą zmianę algorytmów.**

# 01. PKI i podpis cyfrowy

## Problem

Klucz prywatny CA, podpisu kodu lub dokumentu nie powinien pojawić się w pamięci aplikacji jako clear key.

## Przepływ

Dane / hash → aplikacja → CCA Digital Signature Generate → operacja w 4770 → podpis

## Przykładowe usługi CCA

CSNDDSG / CSNFDSG — generowanie podpisu
CSNDDSV / CSNFDSV — weryfikacja
Generowanie pary kluczy i tokeny kluczy PKA

## Dlaczego HSM

Granica dla prywatnego klucza
Kontrola użycia
Możliwy dual control i audyt

## Warunki / ryzyka

Integracja aplikacji z CCA
Polityka algorytmów
HA dla klucza i domeny
Proces odtworzenia

# 02. Ochrona kluczy aplikacyjnych

## Problem

Aplikacja potrzebuje kluczy do szyfrowania, ale administrator OS nie powinien móc ich odczytać ani dowolnie użyć.

## Przepływ

Aplikacja przechowuje secure key token → CCA rozpakowuje go pod master key wyłącznie wewnątrz HSM → wykonuje operację

## Przykładowe usługi CCA

CSNBKGN2 / CSNEKGN2 — generowanie AES/HMAC
Key import / export / translate zgodnie z typem tokenu
Access Control Points ograniczają dozwolone operacje

## Dlaczego HSM

Oddzielenie klucza od hosta
Centralna polityka użycia
Ograniczenie skutków przejęcia konta OS

## Warunki / ryzyka

Nie zakładać „transparentności”
Zmiany w aplikacji / providerze
Inwentaryzacja tokenów
Rotacja master keys

# 03. Szyfrowanie danych i generowanie kluczy

## Problem

Potrzebne są silne klucze, poprawna losowość oraz wykonywanie wybranych operacji poza domeną aplikacji.

## Przepływ

Żądanie RNG / key generation → CCA → 4770 → secure key token lub wynik operacji → aplikacja

## Przykładowe usługi CCA

CSNBRNG / CSNERNG — Random Number Generate
CSNBRNGL / CSNERNGL — dłuższy wynik
CSNBKGN2 / CSNEKGN2 — generowanie kluczy zmiennej długości

## Dlaczego HSM

Sprzętowe źródło losowości
Klucz od początku powstaje w granicy HSM
Spójny cykl życia

## Warunki / ryzyka

HSM nie zastępuje szyfrowania masowego
Dobór: gdzie szyfrujemy, gdzie tylko chronimy KEK
Testy przepustowości i kolejek

# 04. Płatności, TR-31 i EMV

## Problem

Klucze płatnicze muszą być transportowane z zachowaniem atrybutów użycia i kontrolowanego importu/eksportu.

## Przepływ

TR-31 key block ↔ CCA Import / Export ↔ secure key token pod master key → operacje płatnicze

## Przykładowe usługi CCA

CSNBT31I / CSNET31I — TR-31 Import
CSNBT31X / CSNET31X — TR-31 Export
CSNBT31R / CSNET31R — Optional Data Read

## Dlaczego HSM

Format wiąże klucz z metadanymi użycia
HSM egzekwuje politykę i transformacje
Pasuje do środowisk bankowych

## Warunki / ryzyka

Zgodność wariantu TR-31 i profilu
Ceremonie kluczowe
Rozdział ról
Nie każdy system płatniczy używa tego samego API

# 05. Wysoka dostępność i replikacja master keys

## Problem

Awaria pojedynczej karty nie może zablokować podpisu, dekryptowania tokenów ani procesów płatniczych.

## Przepływ

HSM A i HSM B mają zgodne master keys → token utworzony pod MK może być użyty na obu → aplikacja przełącza ruch

## Przykładowe usługi CCA

Rejestry current / old / new
CNM / CNI do zarządzania w środowisku distributed
Procedury weryfikacji i aktywacji master keys

## Dlaczego HSM

Ciągłość działania bez eksportu clear key
Kontrolowane odtworzenie po wymianie karty
Możliwa separacja ośrodków

## Warunki / ryzyka

To nie jest automatyczna replikacja danych aplikacji
Ceremonia i quorum
Testy failover
Synchronizacja rotacji MK

# 06. Kryptografia postkwantowa

## Problem

Organizacja musi zaplanować zmianę algorytmów, zanim istniejące formaty kluczy, aplikacje i protokoły staną się barierą migracji.

## Przepływ

Inwentaryzacja → test obsługi wersji CCA / firmware → generowanie i operacje PQC → integracja z aplikacją → plan migracji

## Przykładowe usługi CCA

CCA 8.4 na 4770: funkcje dla ML-DSA i ML-KEM w udokumentowanym zakresie
Wymagane sprawdzenie platformy, APAR/poziomu kodu i konkretnego verba

## Dlaczego HSM

Sprzętowa granica dla nowych kluczy
Pilot crypto-agility
Ograniczenie migracji „big bang”

## Warunki / ryzyka

Nie obiecywać pełnej gotowości aplikacji
Interoperacyjność protokołów
Wersje OS/CCA/firmware
Wydajność i certyfikacja

# Wdrożenie: minimalny plan techniczny

| | | |
|---|---|---|
| 1 | **Zakres** | Aplikacja, OS, model Power, liczba kart, domeny |
| 2 | **Zgodność** | CCA Support Program, firmware, sterownik, API/verbs |
| 3 | **Governance** | Role, ACP, dual control, ceremonia master keys |
| 4 | **Integracja** | Tokeny kluczy, return/reason codes, obsługa błędów |
| 5 | **HA / DR** | Zgodne MK, przełączenie, wymiana karty, kopie procedur |
| 6 | **Testy** | Funkcjonalne, wydajnościowe, failover, audyt |

**Warunkiem produkcji jest odtwarzalna ceremonia i przetestowany failover — nie tylko działający verb.**

# Rekomendacja: od czego zacząć

## Pilot: ochrona klucza aplikacyjnego + HA

1. Wybierz jeden klucz wysokiej wartości (KEK lub klucz podpisu)
2. Wygeneruj go w HSM i przechowuj wyłącznie secure key token
3. Uruchom tę samą operację na dwóch kartach ze zgodnymi master keys
4. Wymuś failover i odtworzenie po symulowanej wymianie karty
5. Zmierz opóźnienie, przepustowość i obsługę kodów błędów

## Dlaczego ten pilot

- Łączy bezpieczeństwo z ciągłością działania
- Nie wymaga pełnego programu płatniczego
- Daje mierzalne kryteria sukcesu

## Kolejny krok

- TR-31 — jeśli klient ma domenę płatniczą
- PQC — jako osobny tor crypto-agility
- PKI — jeśli priorytetem jest podpis / CA

# IBM Power + 4770 HSM — mapa decyzji

### PKI i podpis

Prywatny klucz nie opuszcza HSM

### Klucze aplikacyjne

Secure token + kontrola użycia

### Szyfrowanie + RNG

Klucz powstaje w granicy HSM

### TR-31 / EMV

Transport z atrybutami użycia

### HA master keys

Ciągłość po awarii karty

### PQC

Pilot crypto-agility

**Wybieraj HSM dla granicy zaufania klucza, kontroli użycia i ciągłości — nie jako uniwersalny akcelerator.**

**3 pytania: jaki klucz chronimy? przed kim? jak utrzymamy operację po awarii?**

# BADAŁEŚ SIĘ?

WKRÓTCE POZNACIE DIAGNOZĘ.

