AWS re:Invent

NOV. 29 – DEC. 3, 2021 | LAS VEGAS, NV

CMP302

Powering next-gen Amazon EC2: Deep dive on the Nitro System

Ben Serebrin
Principal Engineer, EC2
He/Him

Agenda

Nitro overview

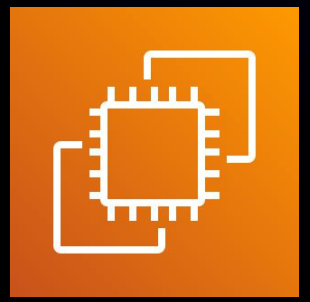
Increasing breadth of offerings with Nitro

Enhancing performance with Nitro

Innovating for instance longevity

Raising customers' security bar

What is Nitro?



Amazon EC2 Nitro

Launched in November 2017

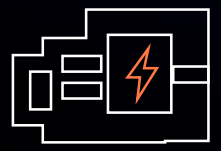
In development since 2013

Purpose-built hardware and software

Custom hypervisor developed for AWS

Every new instance launch uses Nitro

Nitro in three parts



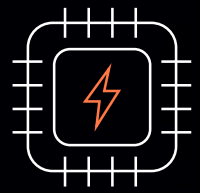
Nitro cards

VPC Networking

Amazon Elastic Block Store (EBS)

Instance storage

System controller



Nitro security chip

Integrated into motherboard

Protects hardware resources

Hardware root of trust

Nitro hypervisor

Lightweight hypervisor

Memory and CPU allocation

Bare-metal like performance

Let’s dig a little deeper into the components

Nitro cards

Nitro card for VPC

- ENA controller
- VPC data plane (encapsulation, Security groups, limiters, routing)

Nitro card for EBS

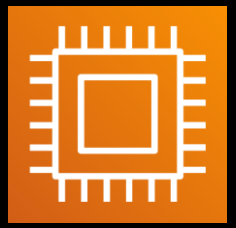
- NVMe controller
- EBS data plane (encryption support, NVM to remote storage protocol)

Nitro card for instance storage

- NVMe controller
- Instance Storage data plane (transparent encryption, limiters, drive monitoring)

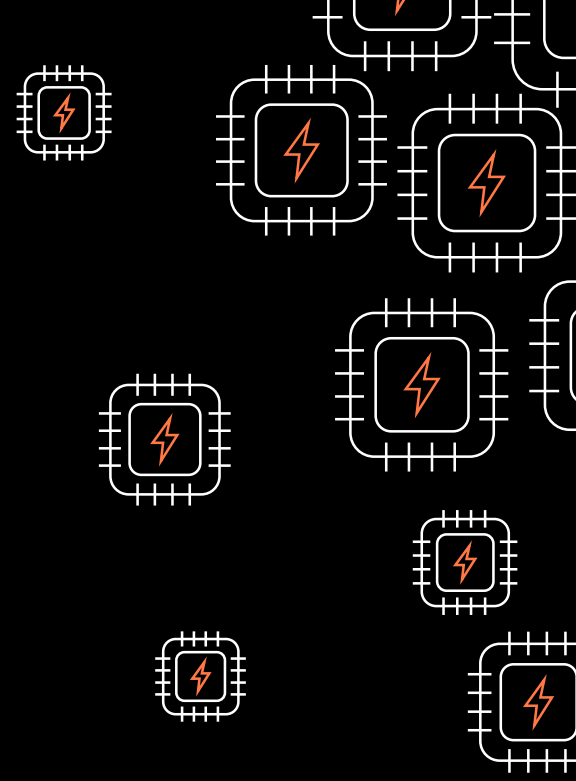
Nitro card controller

- Provides passive API endpoint
- Coordinates all other Nitro Cards, Nitro Hypervisor, and Nitro Security Chip
- Controller Hardware (Root of Trust, provides measurement and attestation)



aws

Nitro security chip



Custom microcontroller that traps all I/O to non-volatile storage

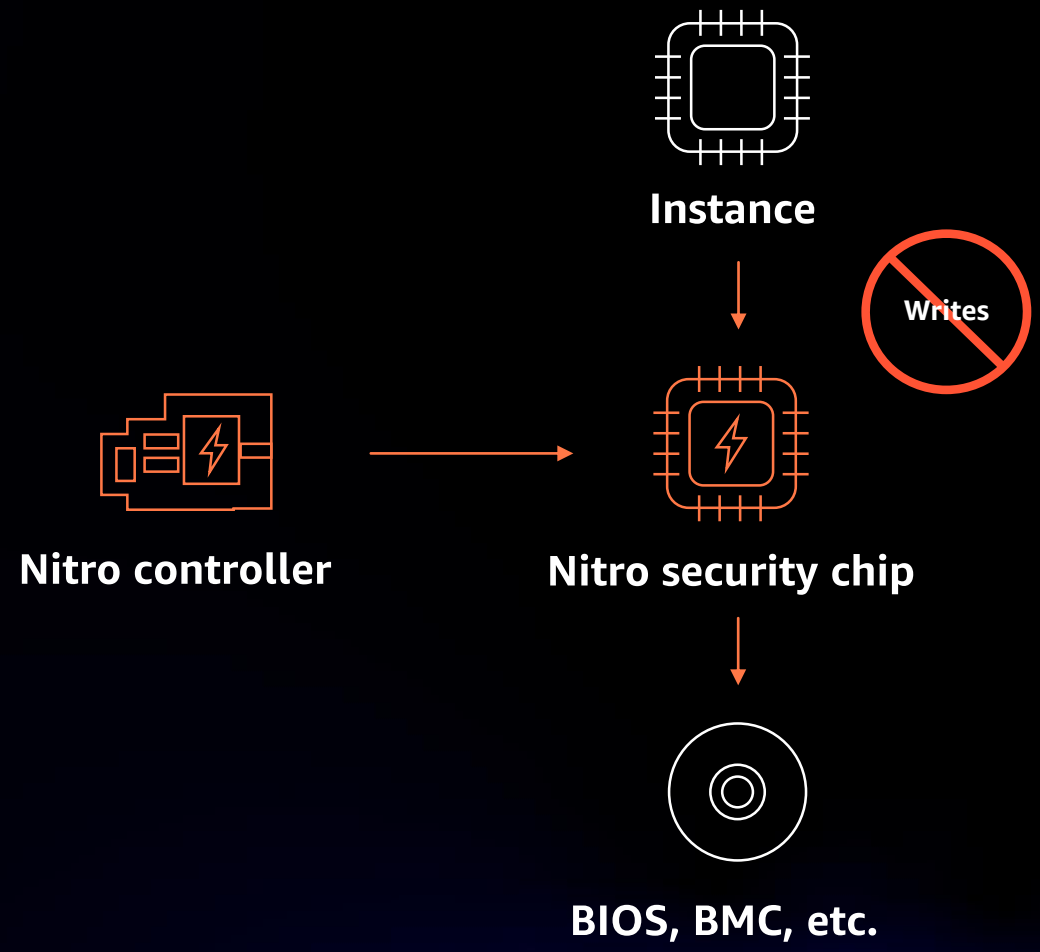
Used by Nitro controller to monitor hardware and validate and update system firmware

Building block of a simple, hardware-based root of trust

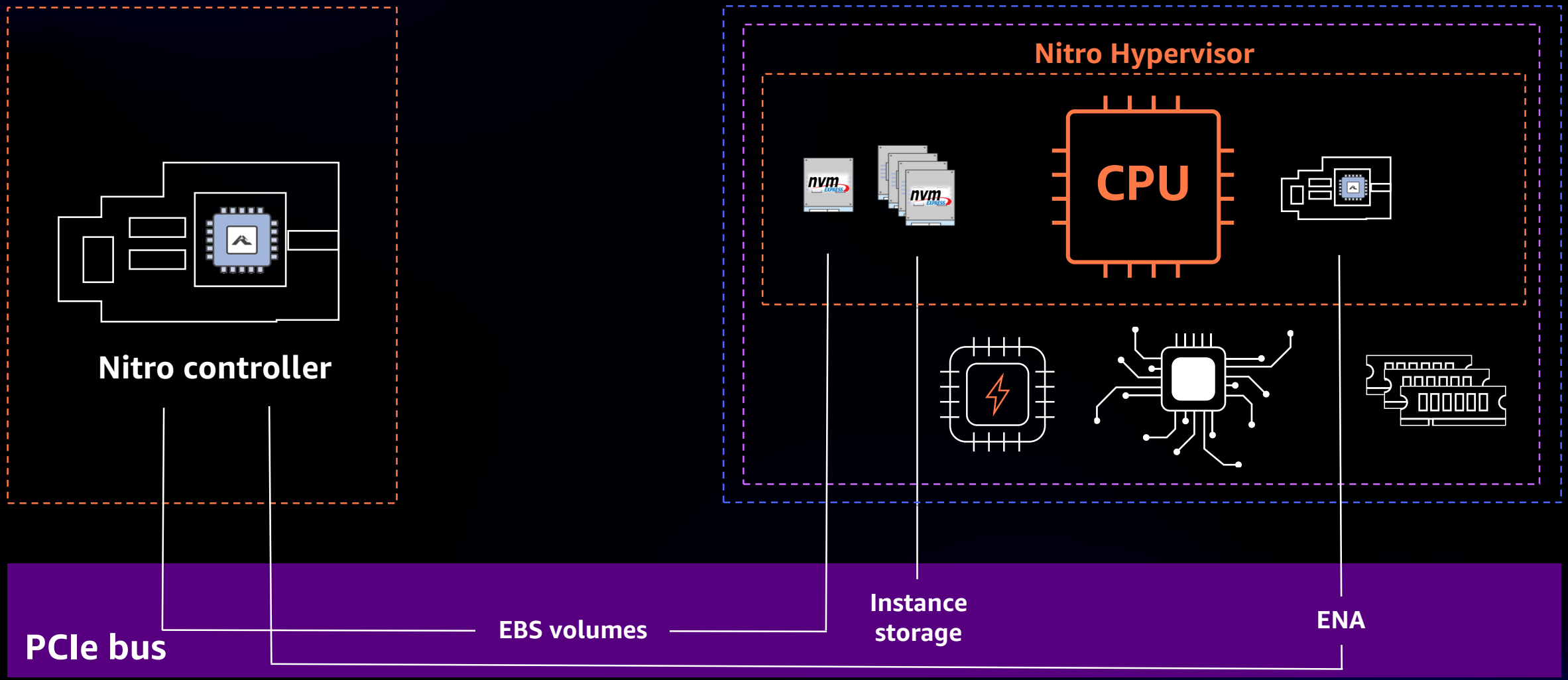
Nitro hardware root of trust

All write-access to non-volatile storage is blocked in hardware

Simplification comes from offloading to cards, no legacy

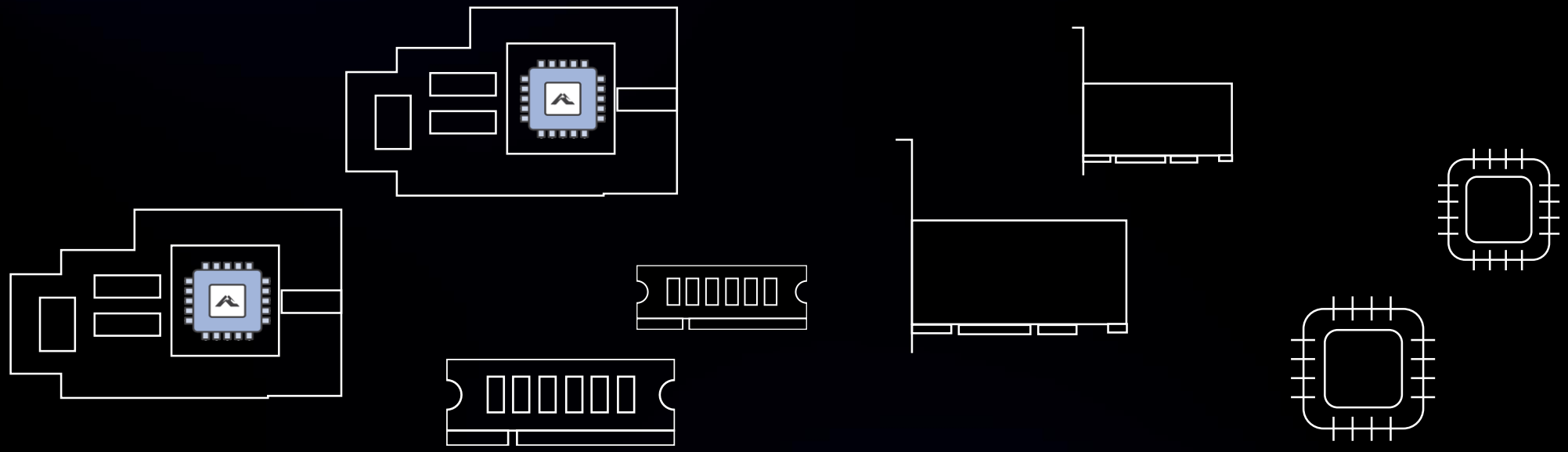


Nitro architecture full view



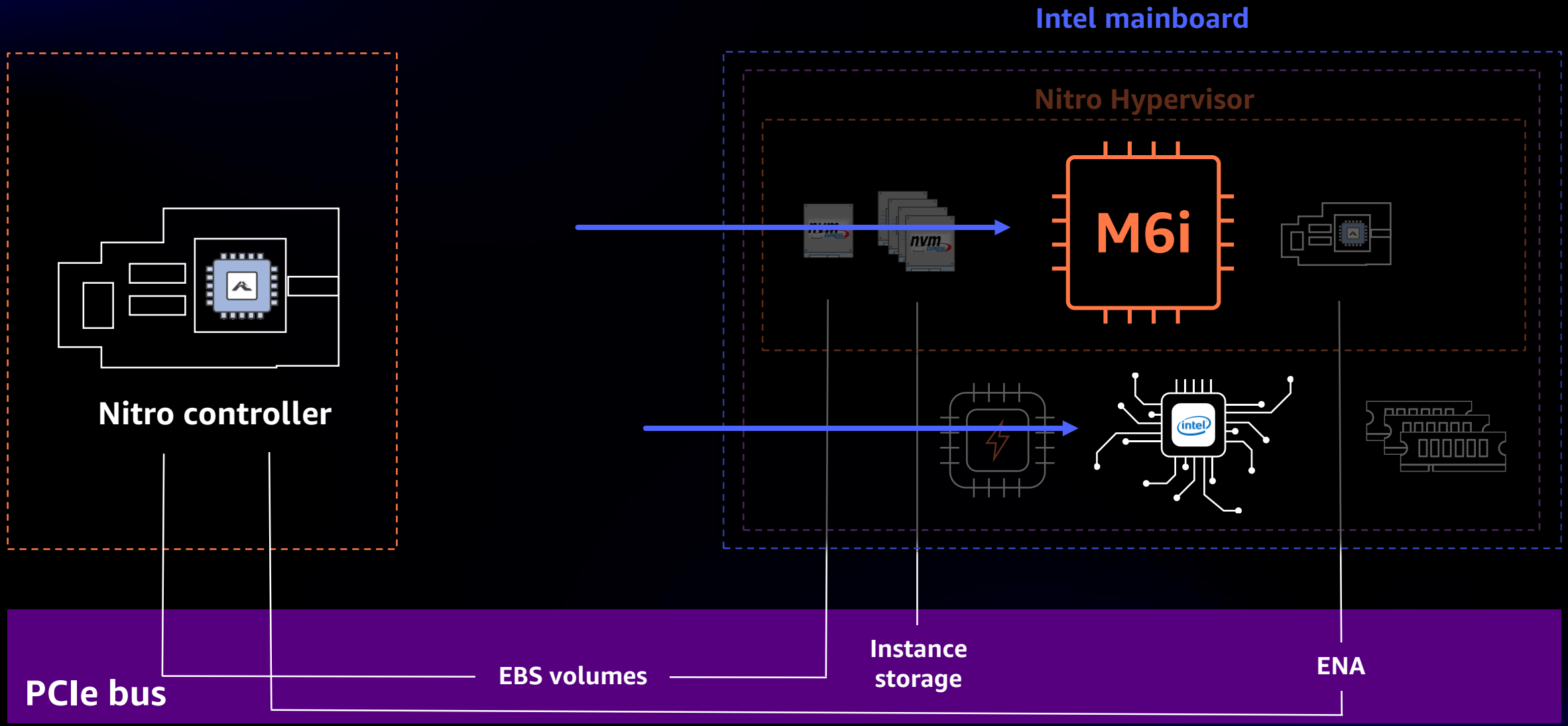
Increasing breadth of offerings with Nitro

Nitro modular architecture

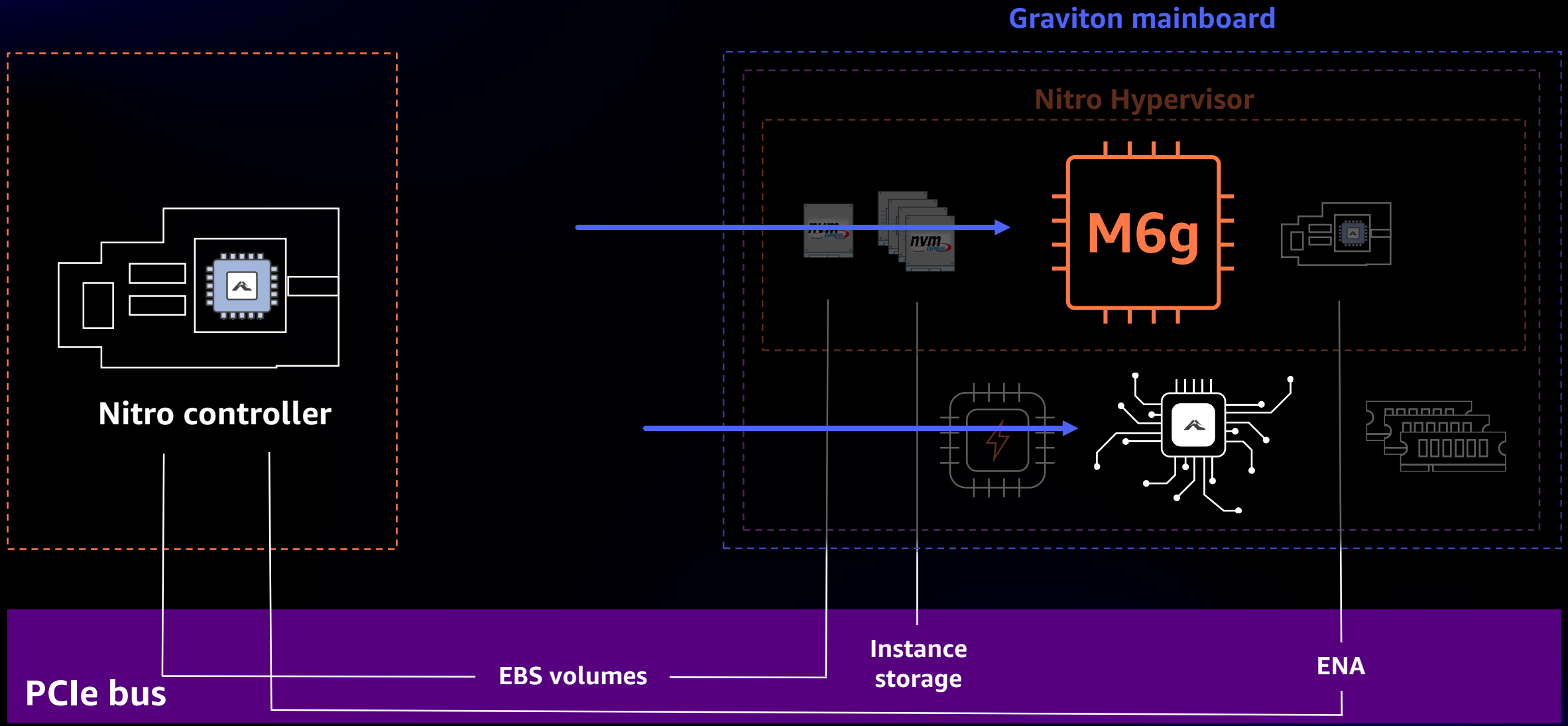


We reuse Nitro components to build a growing variety of systems

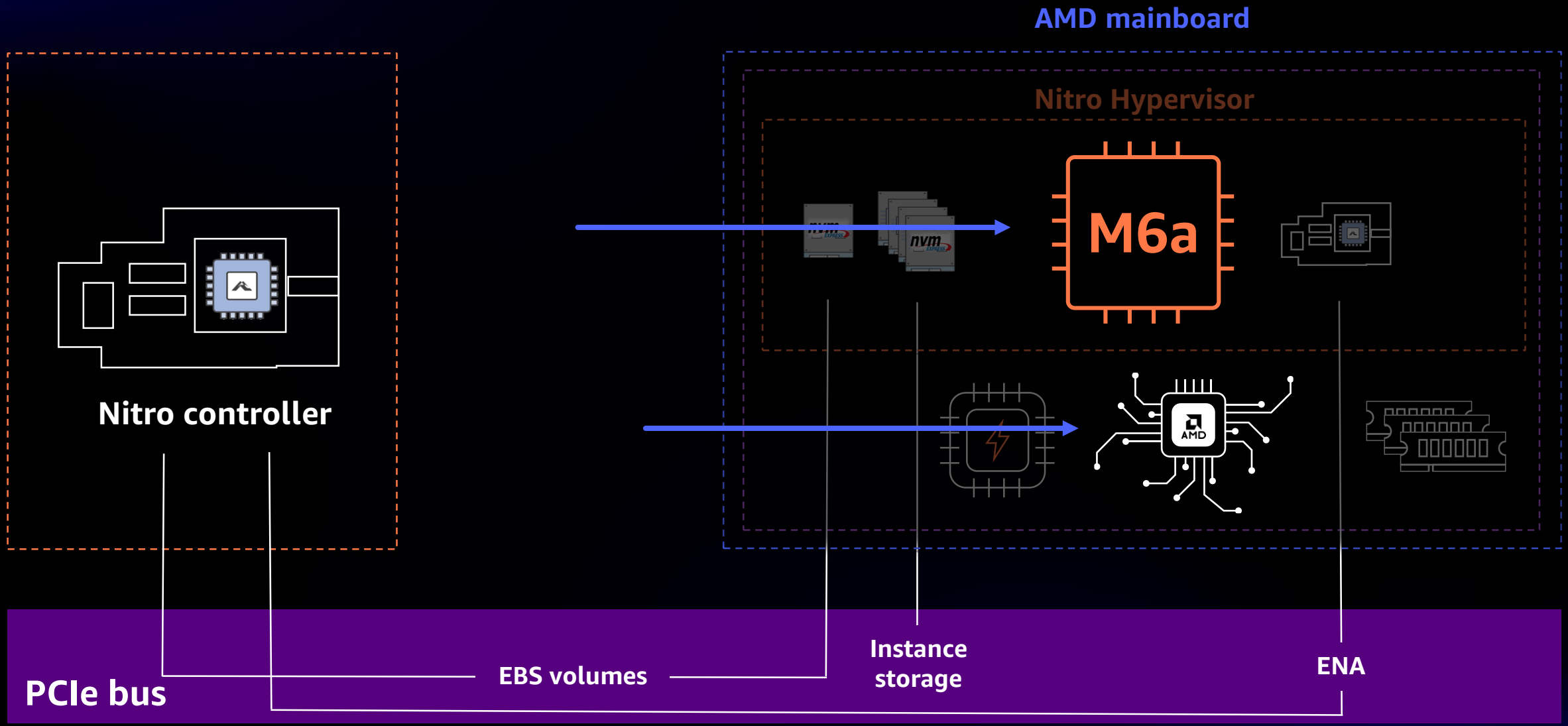
Nitro modularity: CPU choice



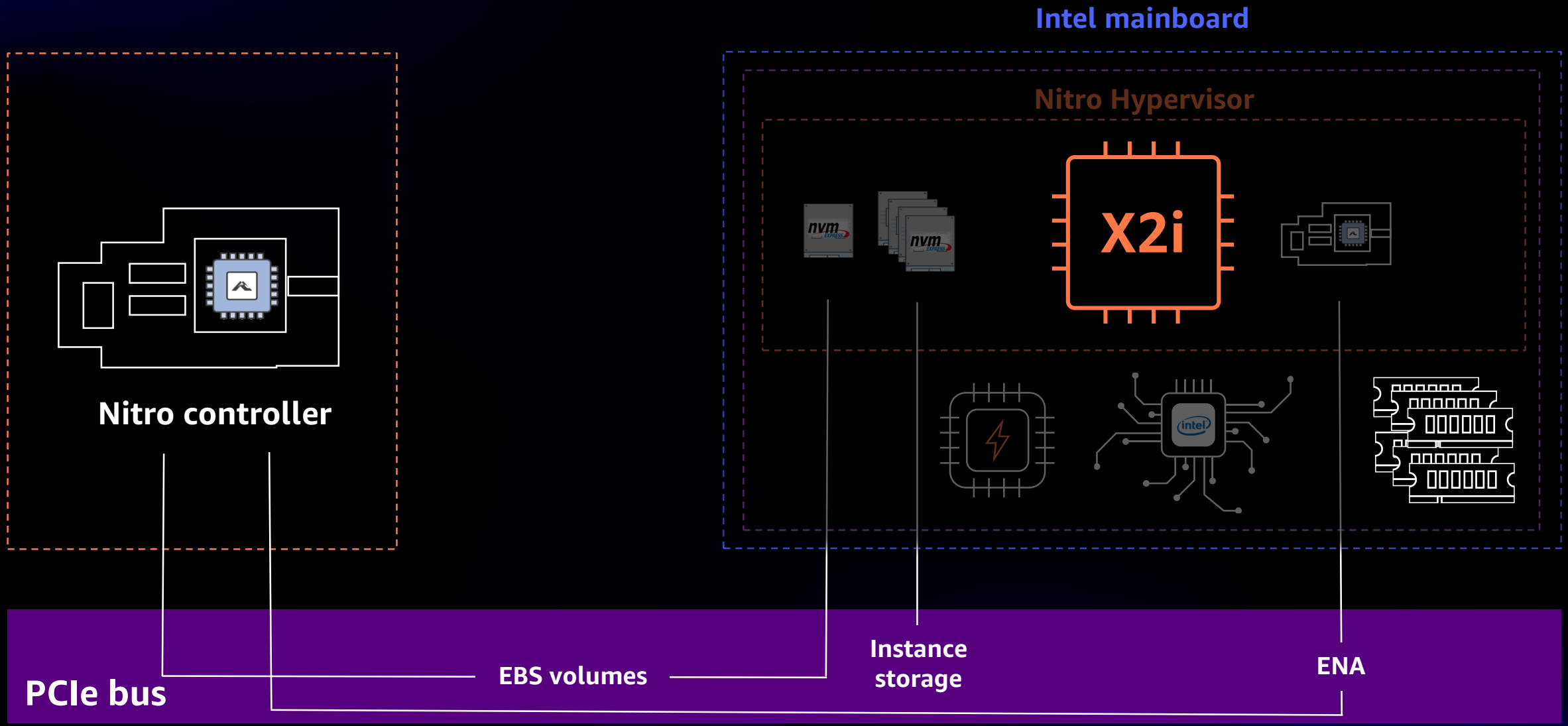
Nitro modularity: CPU choice



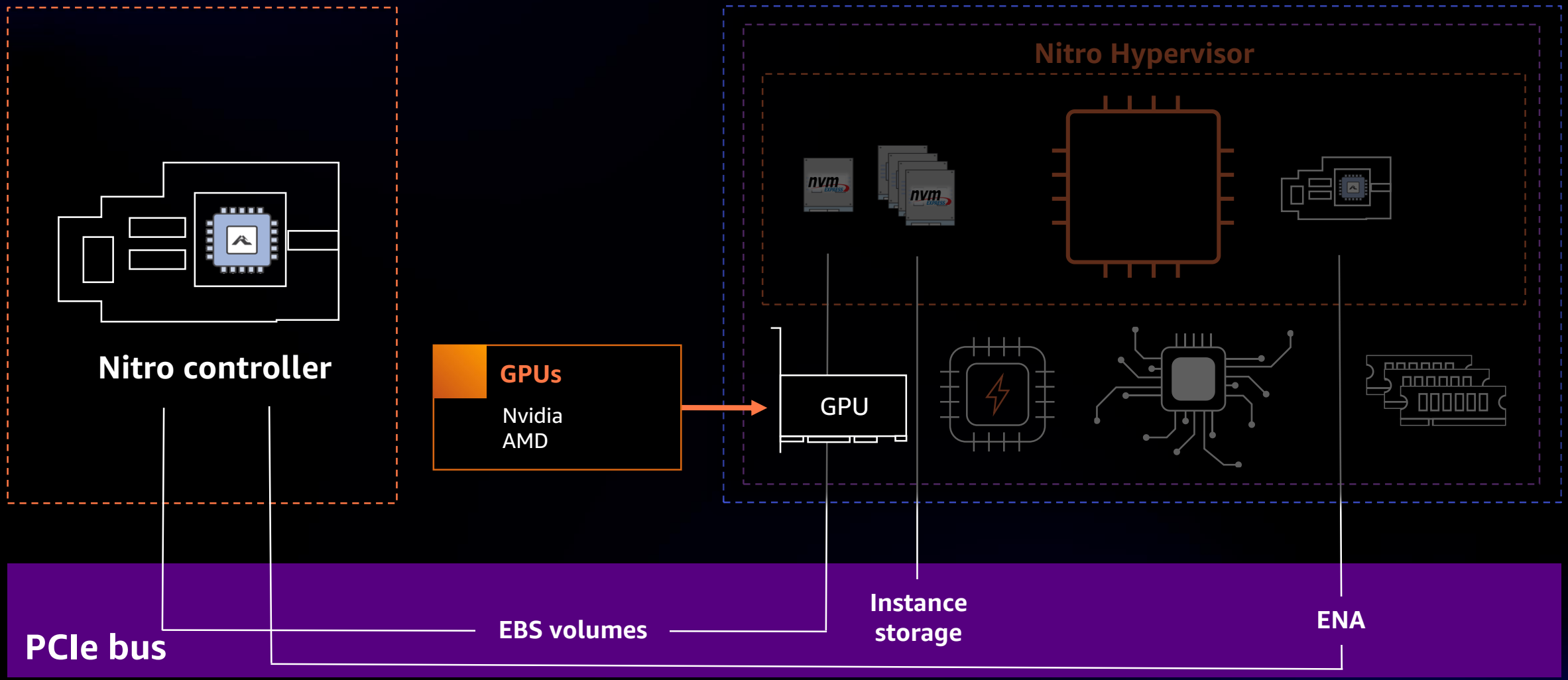
Nitro modularity: CPU choice



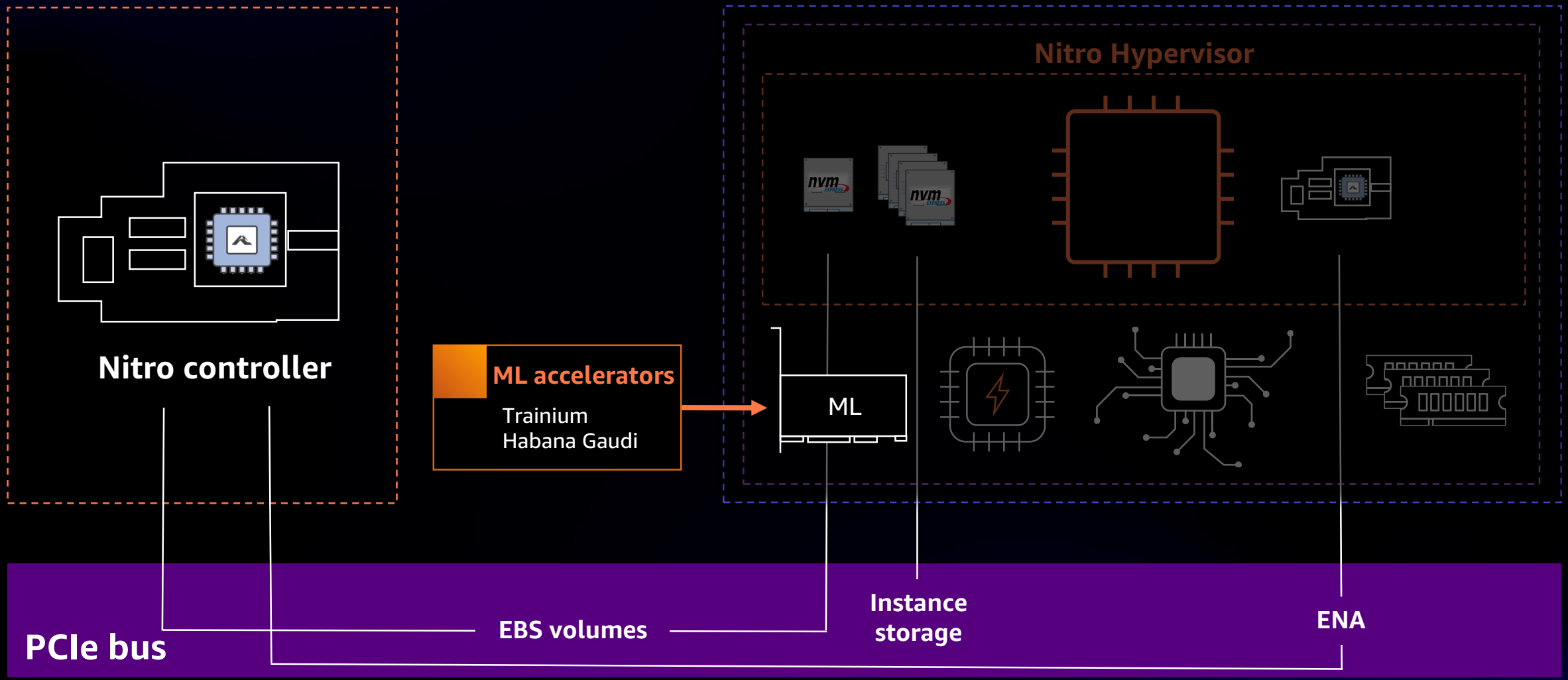
Nitro modularity: Memory



Nitro modularity: Accelerators

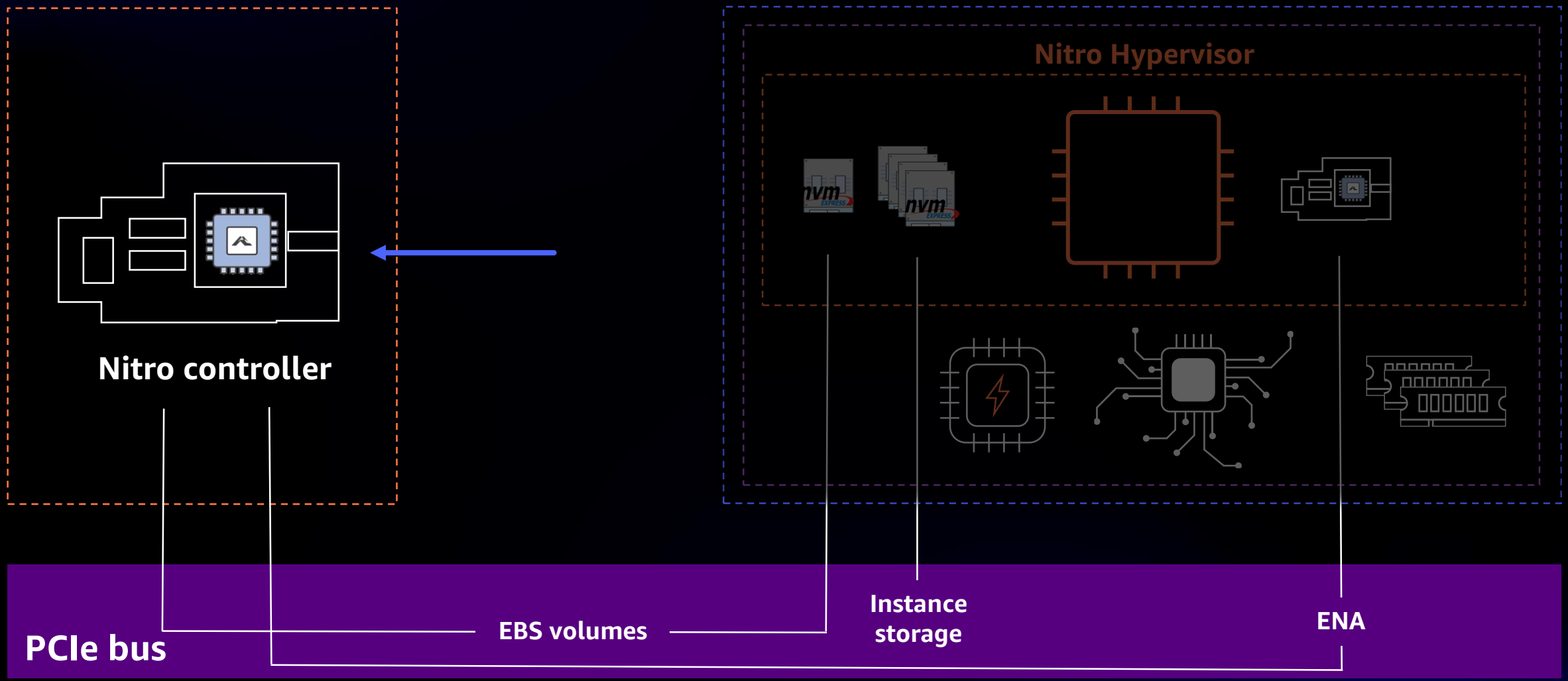


Nitro modularity: Accelerators



Nitro performance growth

Performance modularity for Nitro



Nitro performance evolution: Networking

Upgraded Nitro cards provide increased bandwidth

Customers can upgrade ENA performance without driver updates

Example instance family	M1	C3	C5	C5n	P4d
Maximum network bandwidth (Gb/s)	1	10	25	100	400

Nitro performance evolution: Storage

MODULARITY AND SCALE YEAR-OVER-YEAR

Amazon EBS storage bandwidth tuned for customer demand

Example instance family	R5	U1	R5b
EBS bandwidth (Gb/s)	19	38	60

Nitro performance evolution: Local SSD

CUSTOM SSD OPTIMIZED FOR LATENCY

NitroSSD addresses customer latency stability needs for predictable performance

Example instance family	I3	I4 (Nitro SSD)
Local SSD Latency	baseline	-60%
Local SSD Jitter	baseline	4x lower
Local SSD Capacity	15TB	30TB

Speeding up Nitro lifecycle events

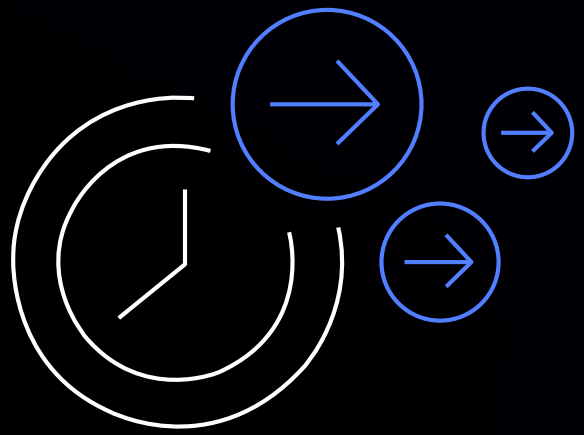
Bare Metal instances

Metal instances allow customers **direct access** to hardware

But long metal instance launch times **do not match** virtual instance times

Customers have needed to **work around** launch times

Accelerating Metal launch



Recent Nitro optimizations speed up Bare Metal instance launches

86% reduction in typical metal instance launch time

In many cases, launch in less than **1 minute**

Accelerating Nitro firmware updates

We keep our systems updated with regular software updates

Instances keep running during Nitro updates

Some customers' workloads are sensitive to performance pause during updates

We introduced a smart management of state during updates

85% reduction in pause time

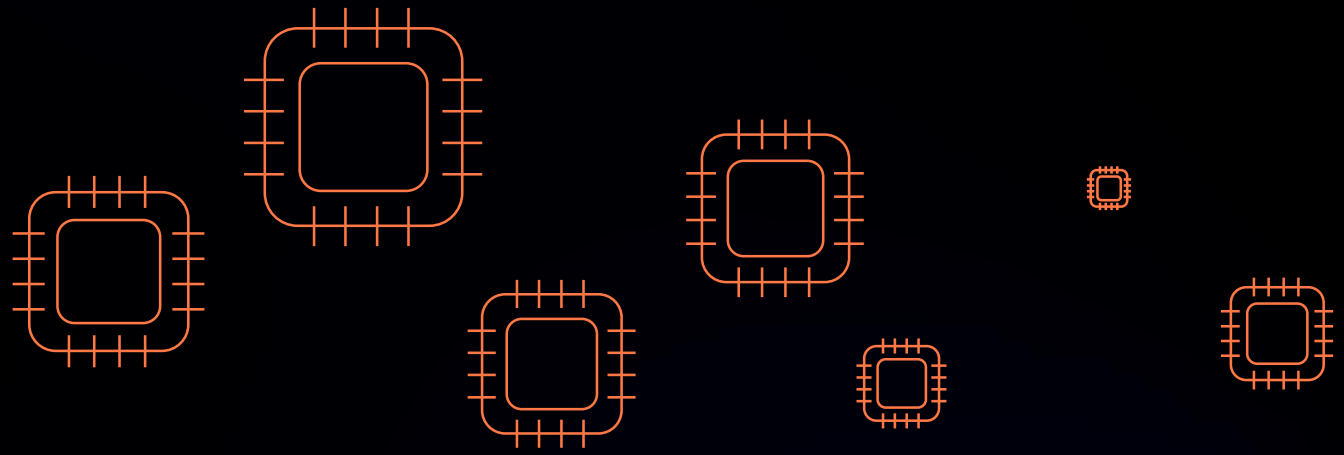
Sub-second pauses for C/M/R instances

Nitro innovation for long-lived instances

Instances outlive their original hardware

Previous generation instances meet many customers needs

Millions of instances run on Xen Hypervisor and end-of-life hardware

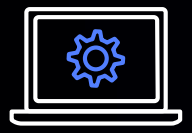


M1.small launched in August 2006, many more since then!

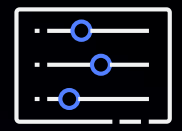
Addressing previous generation instances

Hardware for many instances has reached end of life, but EC2 does not want to force customers to change

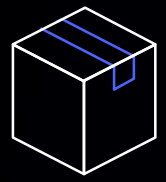
Modern instances differ vs. Xen instances:



Virtual NIC



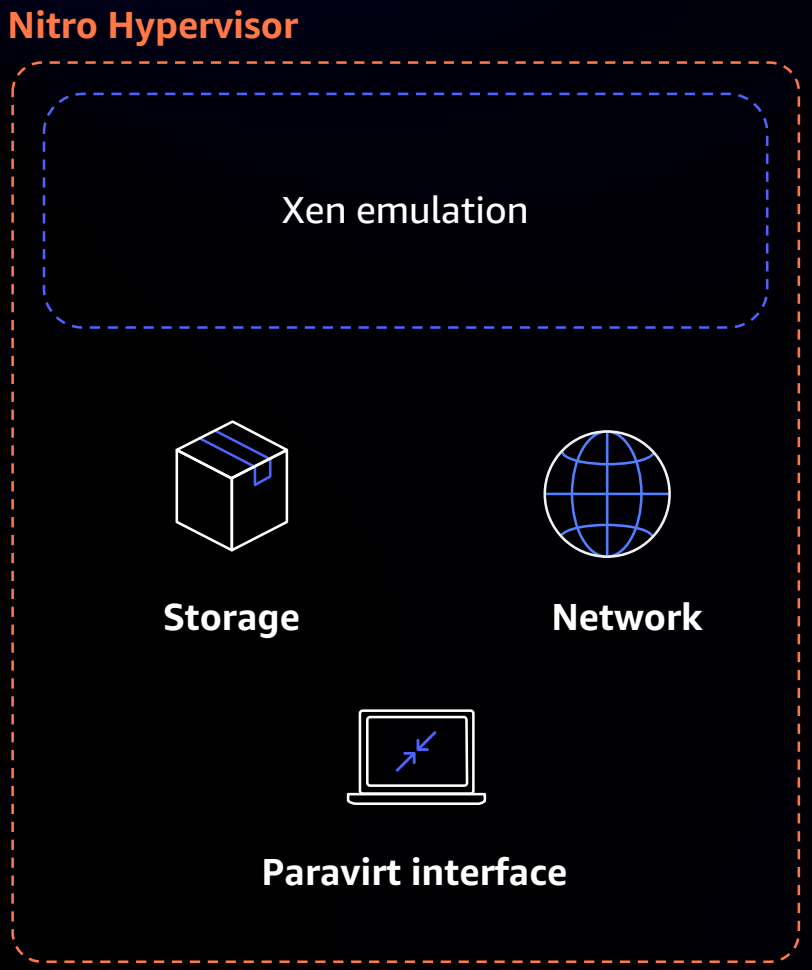
Control plane



Storage device

Network configuration

Innovative solution for older instances



We built Xen emulation in our Nitro hypervisor

A small layer translates to Nitro interfaces; customers do not change their drivers or OS

Customer benefits of Xen on Nitro

Security

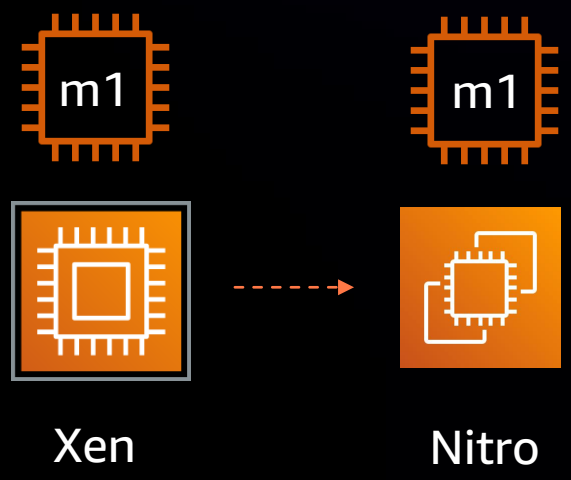
Converging on Nitro for strong isolation and security updates

Stability

Refreshed hardware with new components

Fewer maintenance events

Migration to new hardware

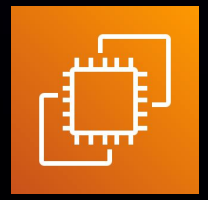


Part of the standard AWS maintenance process: 2-week notice reboot for scheduled maintenance

AWS chooses target hardware; intention is each instance migrates only once. New and rebooted instances can now land on Nitro hardware.

Raising the instance security bar

Hardware and software protection layers



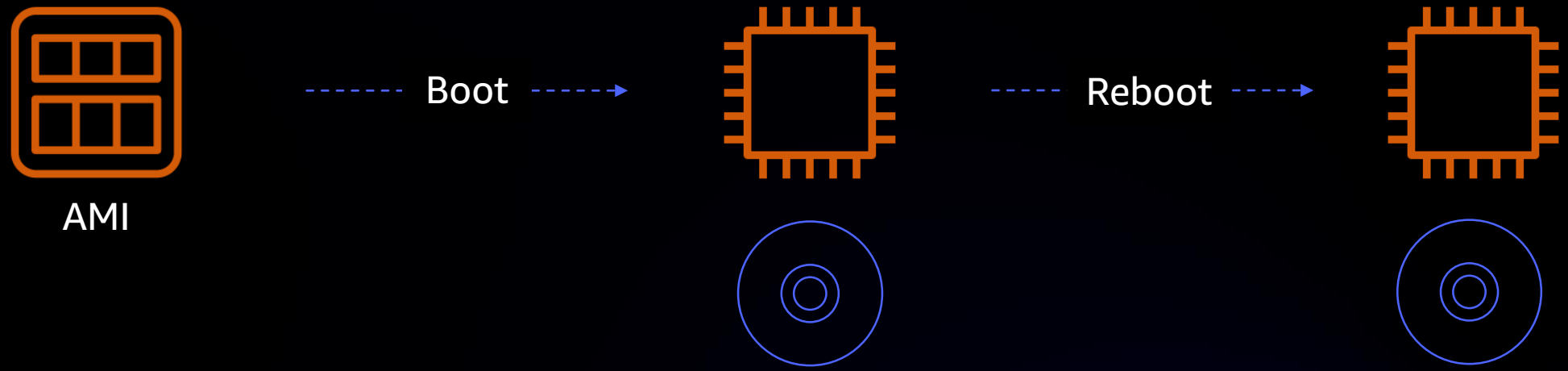
Nitro security chip protects firmware and software

Nitro software is signed and validated

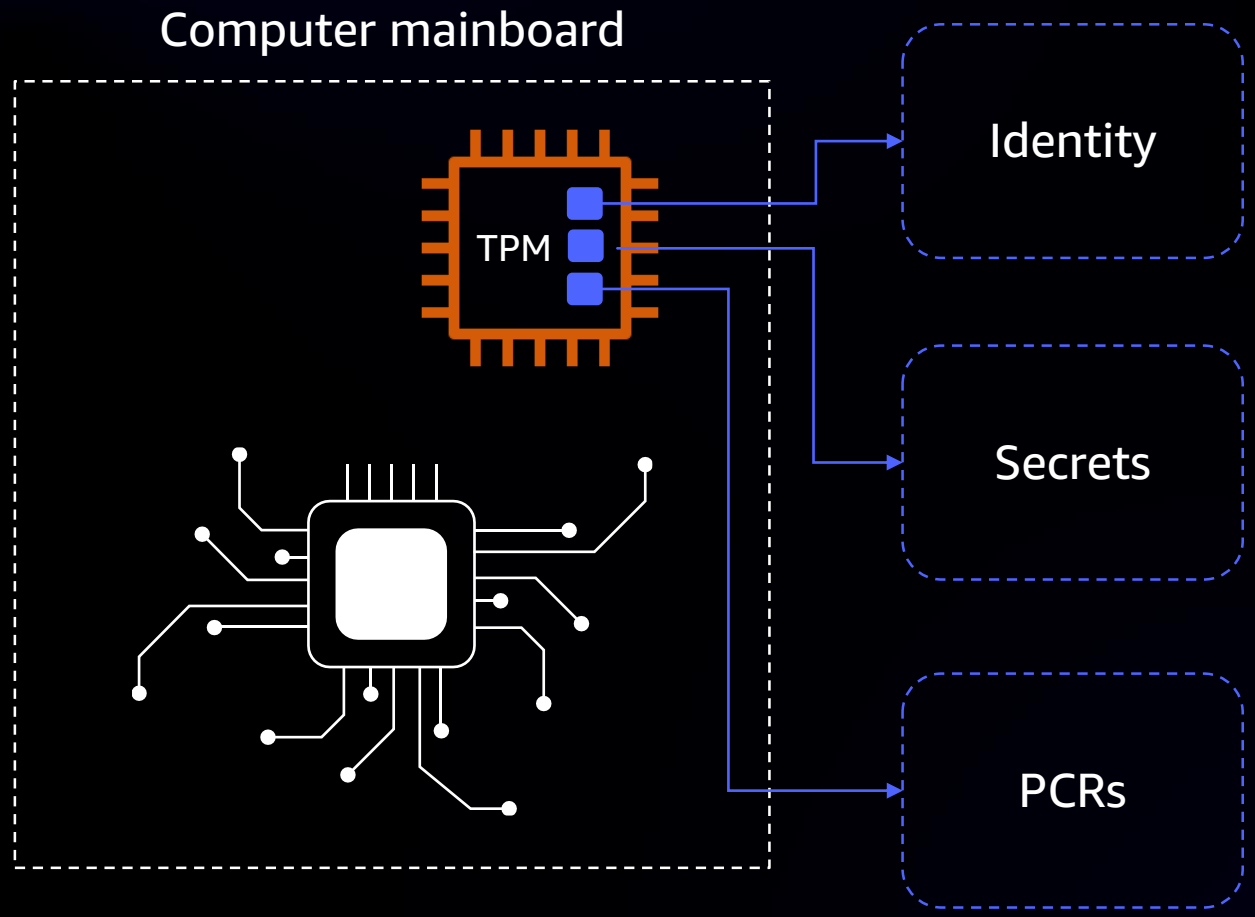
UEFI Secure Boot and NitroTPM enable virtual and metal instances to keep their binaries secure

Secure instance boot

Customers need ways to detect tampering against the guest image and only allow boot if the image is **unmodified**



Deep-dive on TPM



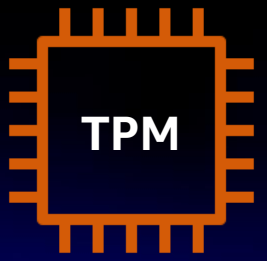
Trusted Platform Module (TPM) is small and isolated, holds small cryptographic payloads

TPM: Platform Configuration Registers (PCR)

TPM has 24 PCRs that contain hashes that build an unmodifiable history

Can be moved forward (“extended”) but not reversed or written

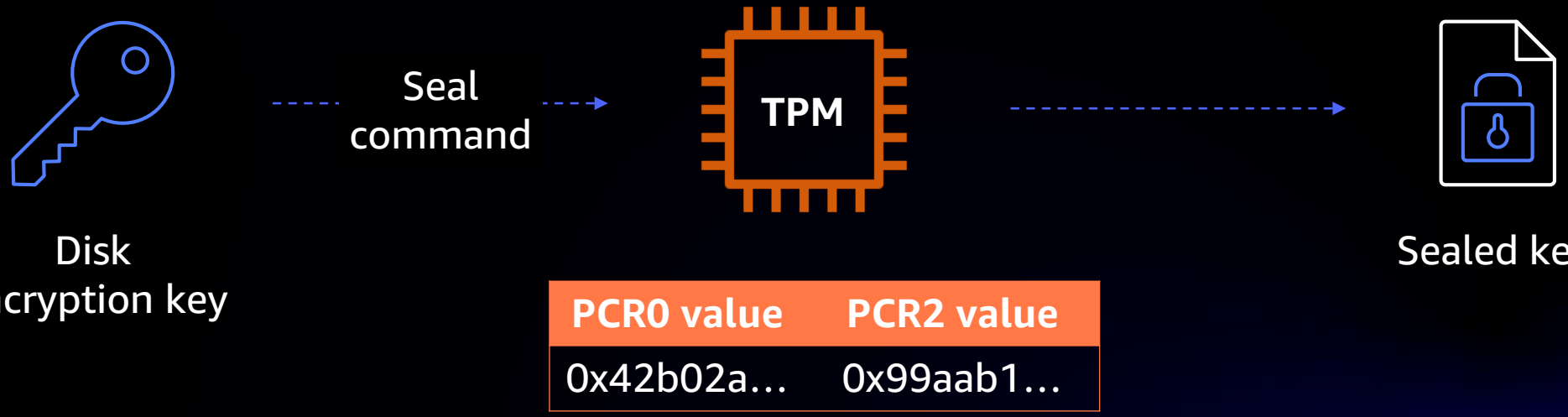
PCR0 value	Operation



Using PCRs to protect secrets

TPM can encrypt secrets for storage, and later un-encrypt them if certain conditions are true: “Sealing” to specific PCR values

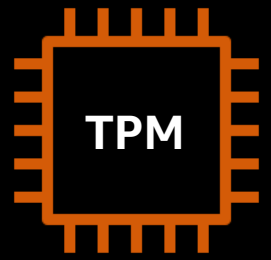
Now only the boot sequence that generates PCR values that match the policy can obtain the disk encryption key



From physical TPM to NitroTPM

LIFT AND SHIFT

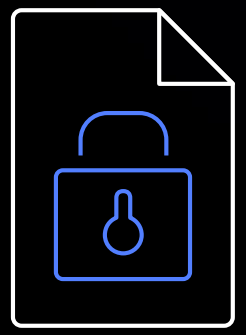
Coming soon: TPM2.0 with NitroTPM for new instance launches



Enables new use cases in EC2:

- Microsoft BitLocker
- DM-Verity
- Attestation
- Linux Unified Key Setup (LUKS)

UEFI Secure Boot



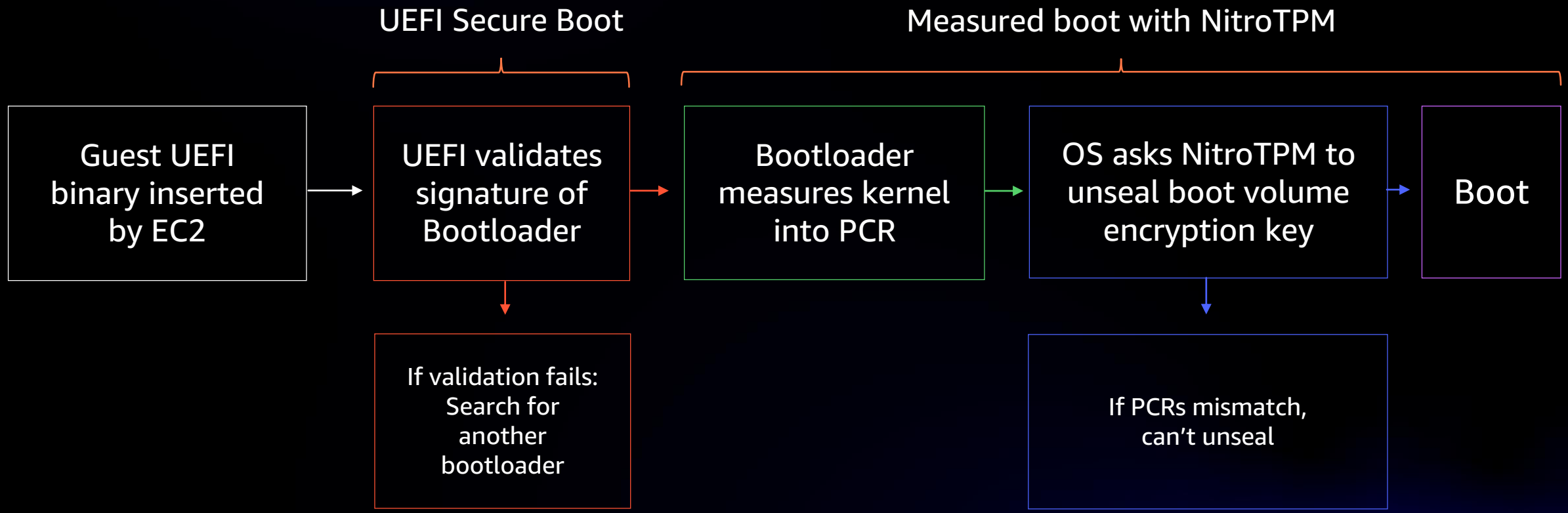
UEFI Secure Boot flow ensures that the boot loader is properly signed by a known authority

Validate the signed bootloader (eg. Grub2) against certificates stored in UEFI

Fall back to backup bootloader or stop if validation fails

Putting it together: Secure instance boot

Drive protection mechanisms Bitlocker and Linux dm-verity can only unlock filesystem if TPM has measured an acceptable boot sequence



What’s next?

Thank you!

Ben Serebrin

serebrin@amazon.com