backend.AI | lablup

Feature Highlight

Confidential AI on Backend.AI

Your Model and Your Users' Data, Hidden from the Host

Why your model is exposed on someone else's GPU servers

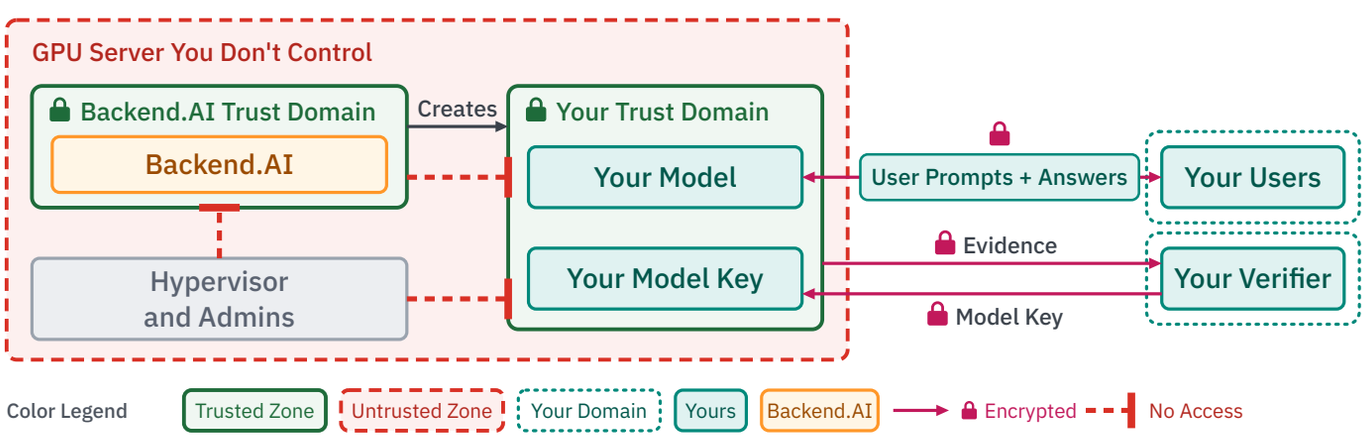
Anyone who copies your model's weight files can run your model without your permission. Your users' prompts and answers can contain personally identifiable information (PII) and source code. Yet you often run the model on GPU servers that a cloud provider or another organization operates. An administrator of those servers, or an attacker who has compromised the hypervisor, can read the servers' memory and disks.

You already protect data on disk with disk and file encryption, and data on the network with TLS. While the model runs, its weights have to be decrypted in memory. Without confidential computing, no technical measure prevents the operator of hardware you don't control from copying the decrypted weights. Today you rely on your contract with the operator instead.

What is confidential computing

Confidential computing protects data in use. With hardware-level security technology such as Intel TDX, your workload runs in an isolated, memory-encrypted environment, with its GPU in confidential mode. The host can start and stop your workload but can't read its memory. Confidential computing also guarantees integrity. Your trust domain checks every container the host asks it to start. If the host changes the command, the settings or the image, the trust domain refuses to start it before your key is ever requested.

Confidential AI with Backend.AI



Backend.AI runs each confidential workload in its own hardware-protected trust domain, with one whole NVIDIA GPU in confidential mode. You ship your model as an encrypted container image and keep its key in your own key broker, which releases the key only after it has checked the hardware evidence. Your users' TLS connections end inside the trust domain, and before sending any data, your users can check what is running there.

Launching a confidential workload

You launch a confidential workload the same way you launch any other. Use your encrypted image and add one resource entry, `confidential.tdx`. Backend.AI then selects the confidential runtime, places the workload on a server that supports it and attaches a GPU in confidential mode.

With confidential computing

```
bai session enqueue '{
  "session_name": "vllm-serve",
  "image_id": "<encrypted vllm/vllm-openai image UUID>",
  // ...
  "resource_entries": [
    // ... cpu, mem and cuda.device as usual
    {"resource_type": "confidential.tdx", "quantity": "1"}
  ]
}'
```

Built on the industry standard

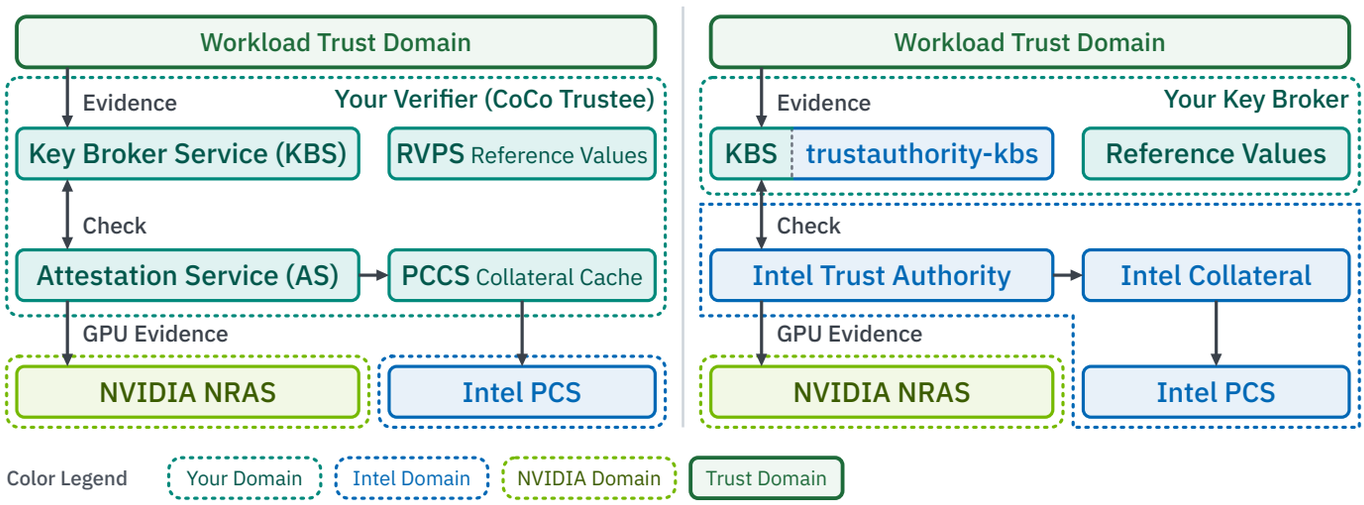
Confidential Containers underneath

Backend.AI builds on Confidential Containers (CoCo), the open-source CNCF incubating project backed by Intel, AMD, NVIDIA, IBM and Microsoft Azure, among others. Backend.AI uses the same components as other CoCo deployments: Kata Containers, the CoCo guest components inside each trust domain, and Trustee as the key broker. Your security team can review an open-source stack they may already be familiar with.

What Backend.AI adds

On Kubernetes, you often build a confidential serving stack from several CoCo operators and a separate model-serving product. Backend.AI uses the same CoCo components without a Kubernetes cluster. Its own scheduler and node agent place confidential workloads directly on bare-metal servers, and the Backend.AI proxy forwards your users' traffic to those workloads. You deploy a confidential model as a regular Backend.AI model service, without a second serving product. Backend.AI's own control plane can also run in a confidential VM, which unlocks its encrypted disk only after its hardware evidence is checked.

Backend.AI with Intel®



The problem with running your own verifier

To release a key safely, you need a verifier that checks the hardware evidence. With a self-hosted Trustee, you operate and secure that verifier yourself: besides the key broker, you run an attestation service and a reference-value service. You also keep a current copy of the collateral data that Intel publishes for its platforms. The attestation service requires this data to check the evidence.

Intel Trust Authority

With Intel Trust Authority, you don't run a verifier. Intel Trust Authority appraises the CPU evidence, forwards the GPU evidence to NVIDIA's attestation service, and returns one signed result for both. Before your users connect, you can appraise the model service's current hardware evidence with your own Intel Trust Authority API key. Backend.AI also works with `trustauthority-kbs`, Intel's key broker, which connects to your existing enterprise key manager or hardware security module (HSM).

Roadmap

- Confidential serving across multiple GPUs, on GPU platforms that support multi-GPU confidential mode
- Confidential sessions that span several nodes, with traffic between trust domains encrypted
- Confidential storage, with encrypted volumes for your datasets and model folders
- Support for AMD SEV-SNP and other confidential-computing hardware

Confidential AI on Backend.AI is in development and not yet a released product. The design shown here and in the booth demo is subject to change. Current scope is limited to one whole NVIDIA GPU per confidential workload, a single node and sites with internet access.

backend.AI | lablup

© 2026 Lablup Inc. All rights reserved. Oct 25 Intel, the Intel logo, and other Intel marks are trademarks of Intel Corporation or its subsidiaries. All other trademarks are the property of their respective owners.

github.com/lablup/backend.ai

www.backend.ai