Confidential Containers

Chris Butler
Chief Architect, CTO Organization
chris.butler@redhat.com

Dave Ripani
Solution Architect
rippa@redhat.com

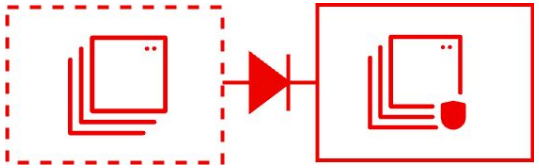
Red Hat

Agenda

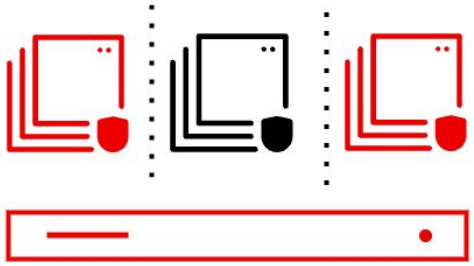
- Confidential Computing 101
- Confidential containers architecture
- Demo 1: Protecting data in memory
- RATS architecture
- Use case scenarios

Focus areas for confidential computing

Multi-level security



Compartmented data



Physical asset compromise

Demo 1: - Showing it protects memory

- Scenario 1: Standard containers (I go to root in the box and find it)
- Scenario 2: kata containers - (e.g. shim qemu) - get in as root; use gcore / find string
- Scenario 3: ‘CoCo’ container - go in dump memory show we can’t find it.

Demo 2:

- **Attestation and releasing secret**
  - **Showing in a TDX container I can get the secret otherwise I can’t**

Deployment arch (high level)

- Multicluster
-

Confidential computing enforces cryptographic protection of applications and data

- storing → at-rest — LUKS, GnuPG,...
- transmitting → in-transit — OpenSSL, TLS,...
- processing → in-use

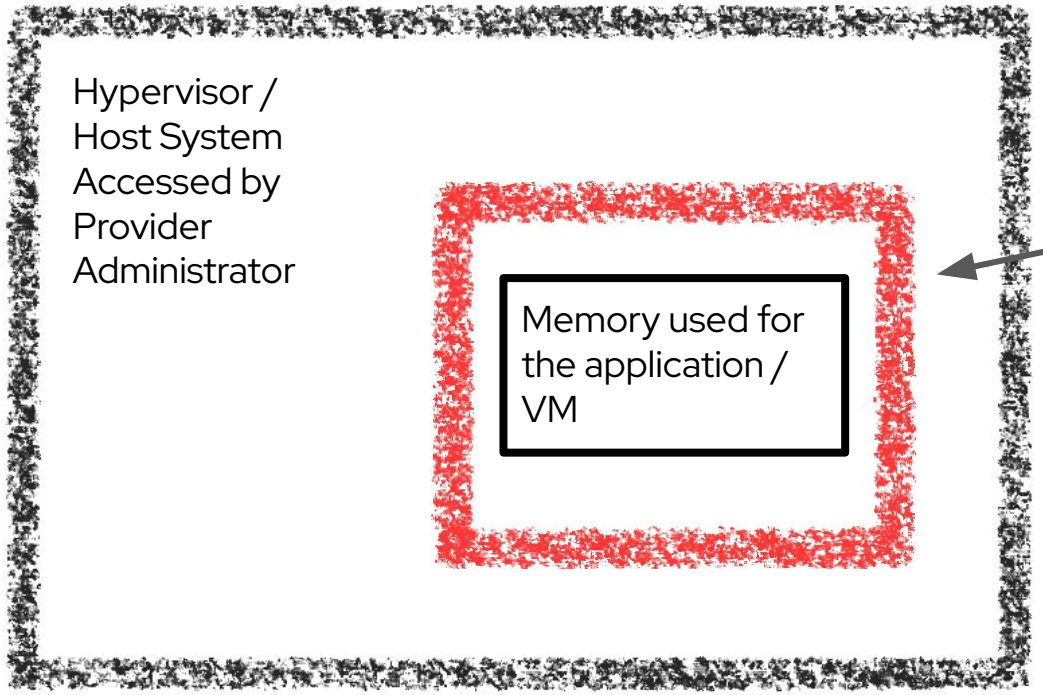
Confidential Computing is about protecting data **in-use**. (incl. integrity)

Enforcing that you do not trust the system admins or infra provider admins

Red Hat

Data in use

- How does Confidential Computing work?



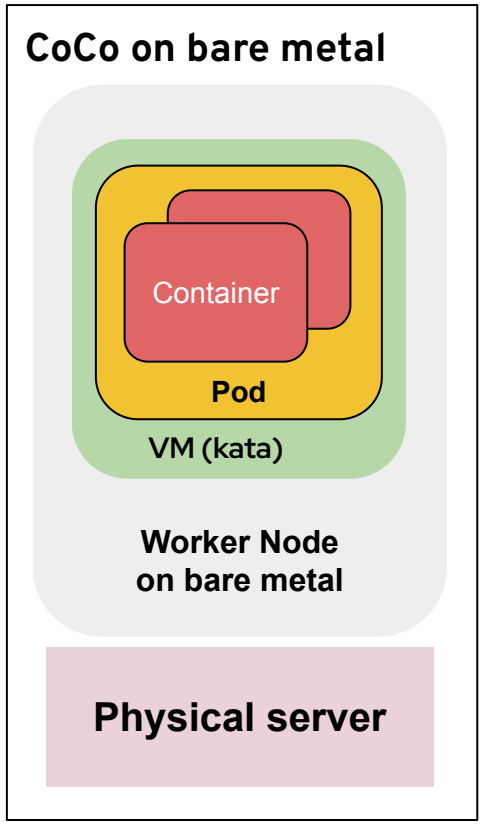
Hidden from the Operating System and Kernel inside a Trusted Execution Environment. Access only through special API.

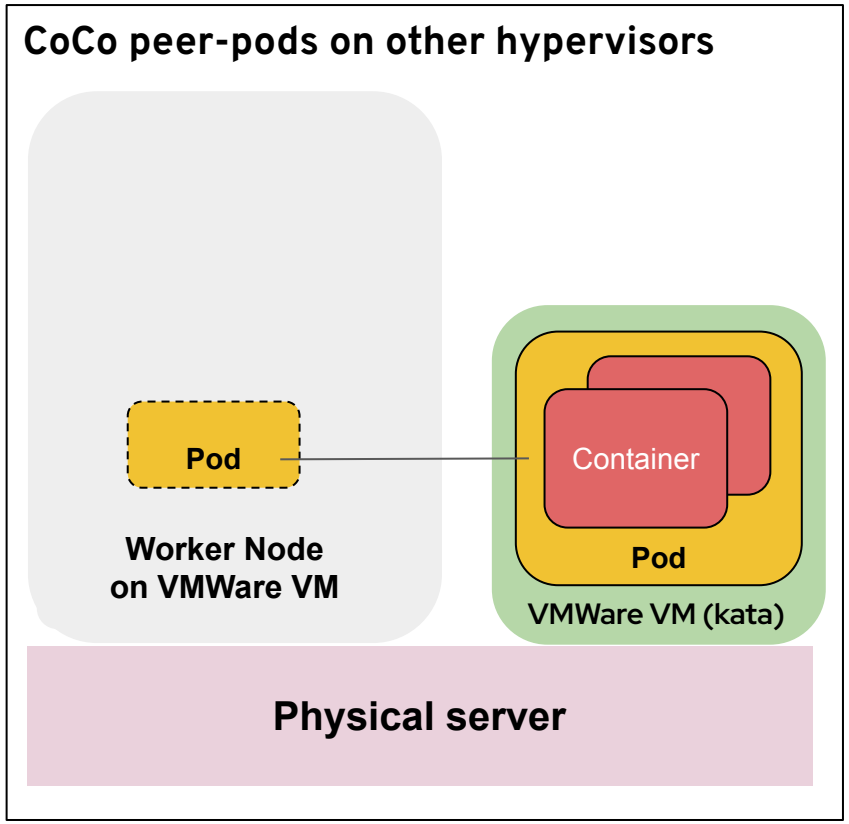
Confidential Containers (CNCF)

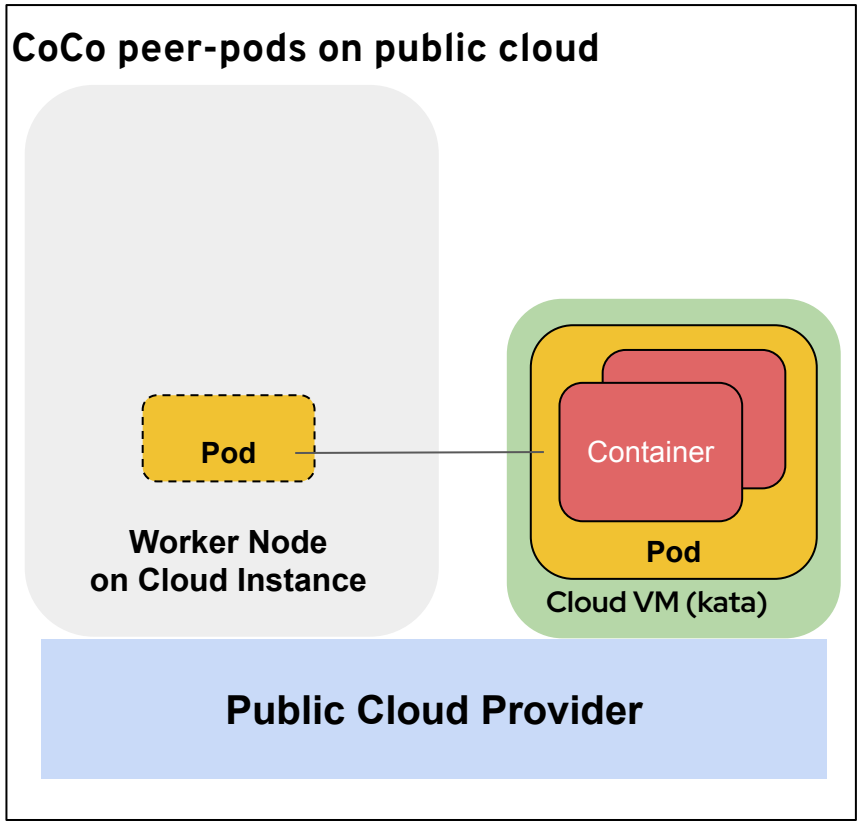
CONFIDENTIAL CONTAINERS

Confidential Containers is an **open source community** working to **enable cloud native confidential computing** by leveraging Trusted Execution Environments to protect containers and data.

Confidential Containers across cloud & bare metal







- Peer-pods evolves the OSC solution from bare metal to hypervisors and public cloud
- Peer-pods code repositories enhance the kata containers project and the confidential containers project

Red Hat

Demo one

Memory isolation in confidential containers

Select a CoCo runtime

Pull data a ‘secret’ in

Do nothing

No special privileges

Store the secret in memory

```yaml
apiVersion: v1
kind: Pod
metadata:
 name: coco-demo
 annotations:
   io.katacontainers.config.hypervisor.default_vcpus: "2"
   io.katacontainers.config.hypervisor.default_memory: "4096"
spec:
 runtimeClassName: kata-cc-tdx
 #runtimeClassName: kata
 initContainers:
 - name: fetch-key
   image: registry.access.redhat.com/ubi9/ubi:9.3
   command:
   - sh
   - -c
   - curl -L https://gist.githubusercontent.com/butler54/21a0.../raw.txt -o /keys/magickey.txt
   volumeMounts:
   - name: keys
     mountPath: /keys
 containers:
 - name: coco-demo
   image: registry.access.redhat.com/ubi9/ubi:9.3
   command:
   - sleep
   - "36000"
   securityContext:
     privileged: false
     seccompProfile:
       type: RuntimeDefault
   volumeMounts:
   - name: keys
     mountPath: /keys
 volumes:
 - name: keys
   emptyDir:
     medium: Memory
```


Red Hat

Threat Vectors:

Pod Images: Risk of tampering/access.

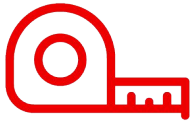
***Pod Memory*: Infrastructure provider access.**

Pod Data: Provider tampering/access.

Encrypted Workloads

+

Remote Attestation

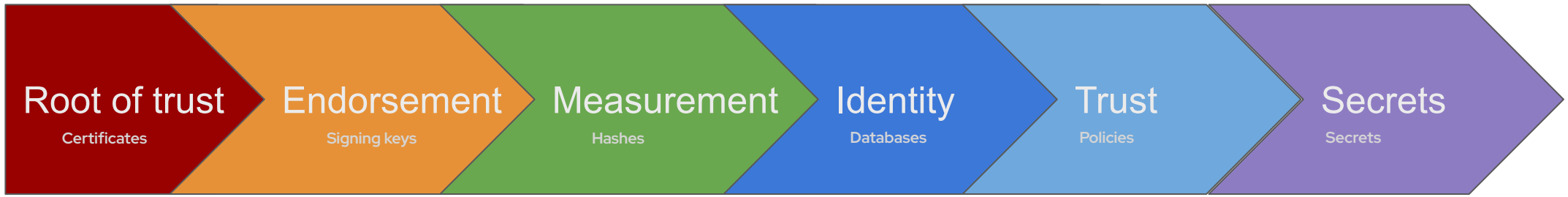


=

Confidential Computing

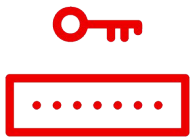
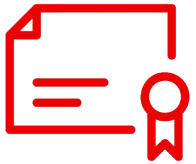
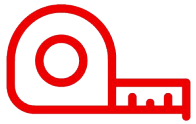
Establishing a chain of trust

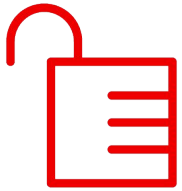
REMITS



Example:

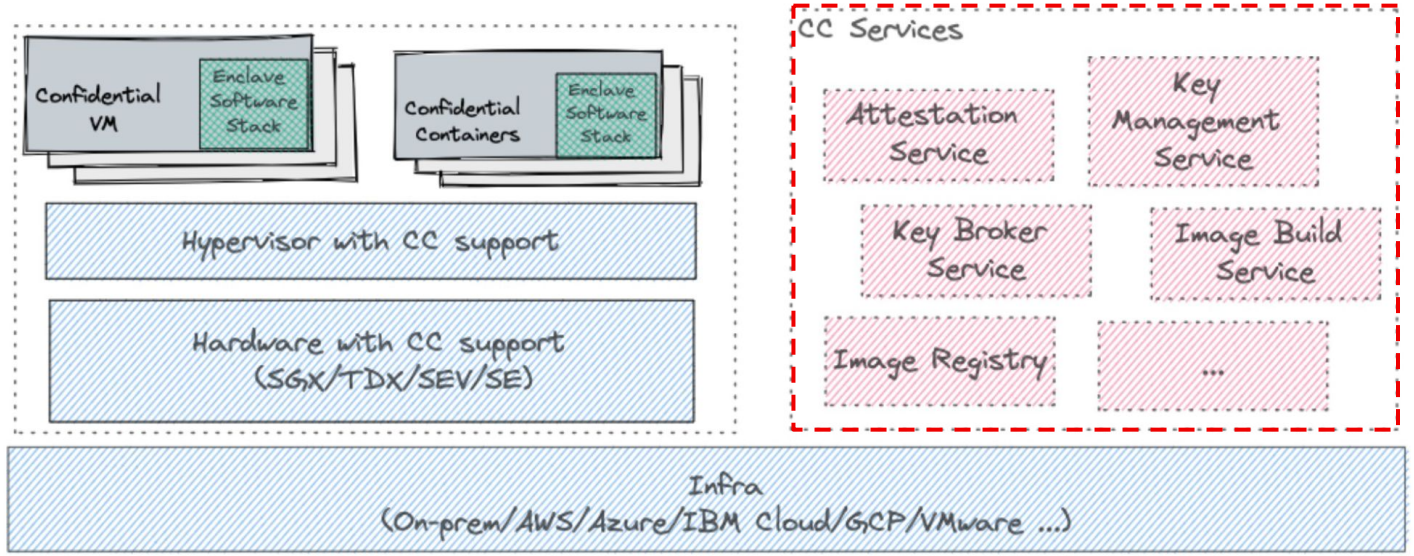
private key from a chip manufacturer	per device authenticity info	crypto hash over code & config	compare evidence with ref. values	apply diff. policies for diff. envs	share secrets, e.g. encryption keys

Red Hat

Confidential Computing Services

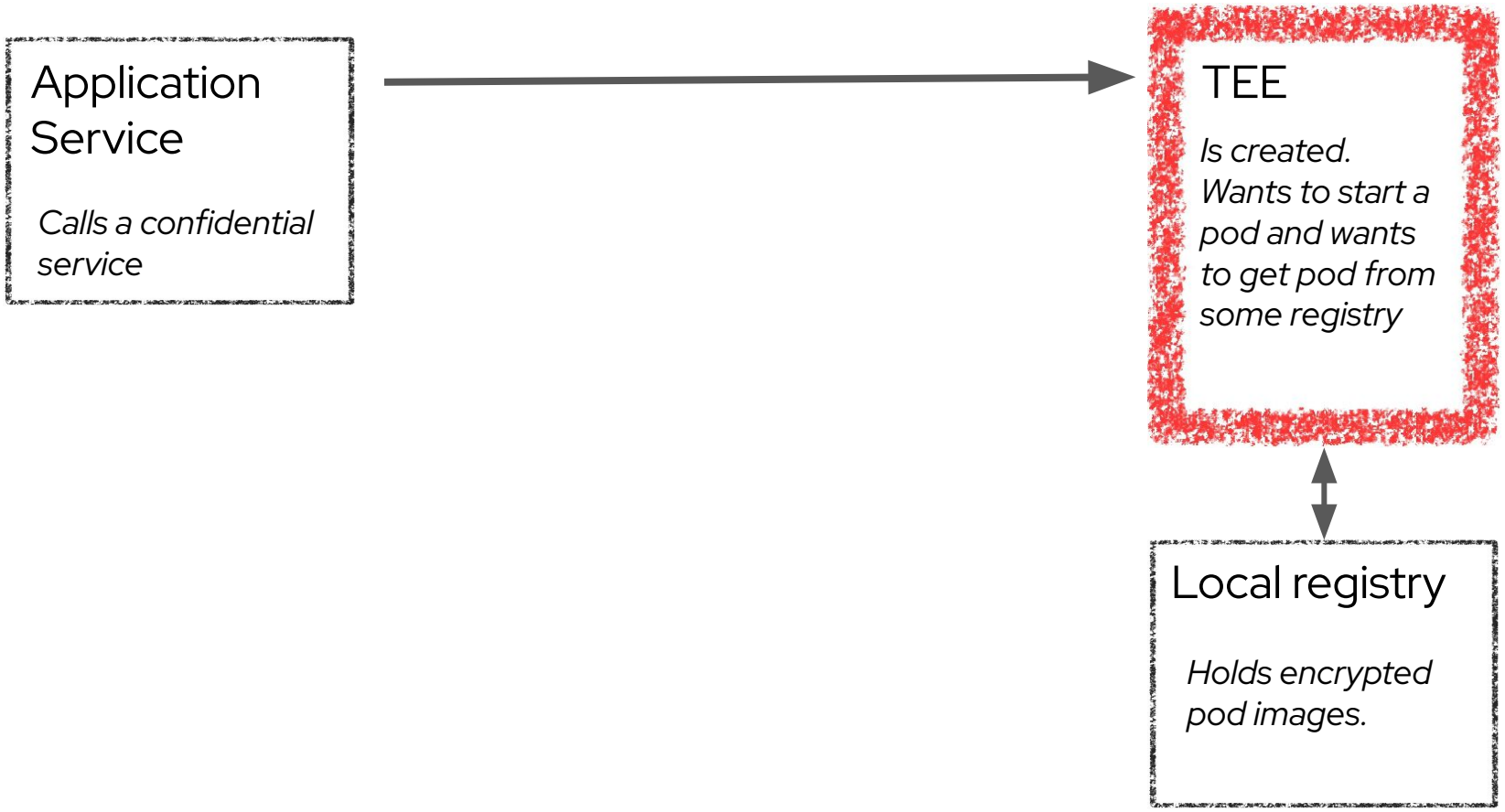


- **Attestation Service** - The primary purpose of the attestation service is to validate the evidence provided by the hardware TEE. This is the *Verifier*, as defined in the RATS architecture.
- **Key Broker Service (KBS)** - The KBS is the *Relying Party*, as defined by the RATS architecture. Following are its primary functions:
  - Receive evidence from the *Attester* (confidential VM or container) via a challenge-response protocol.
  - Relay the evidence to the Attestation Service for verification.
  - Apply appraisal policy for the returned Attestation Results to assess the trustworthiness of the *Attester*.
  - Interact with the Key Management Service to retrieve the keys and then send them back to the *Attester*.

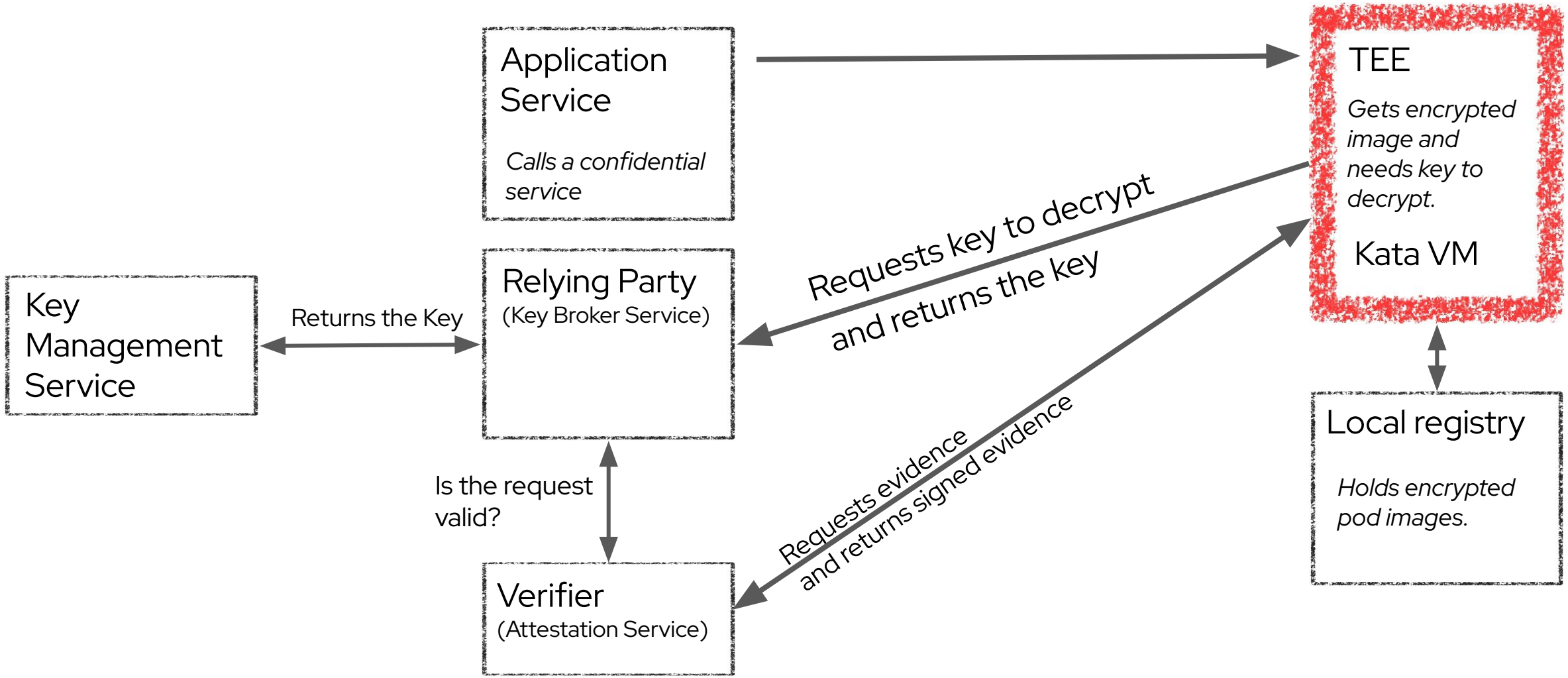
- **Key Management Service (KMS)** - A service for securely storing, managing and backing up of cryptographic keys used by applications and users.
- **Image Build Service** - Services used to build confidential containers or VM images for end users.
- **Image Registry** - A service that is used to store encrypted and/or signed container and VM images required for CC workloads. Examples of such registries include Quay.io, Docker Hub, CSPs provided registries, etc.

https://github.com/confidential-containers/documentation/blob/main/architecture.md

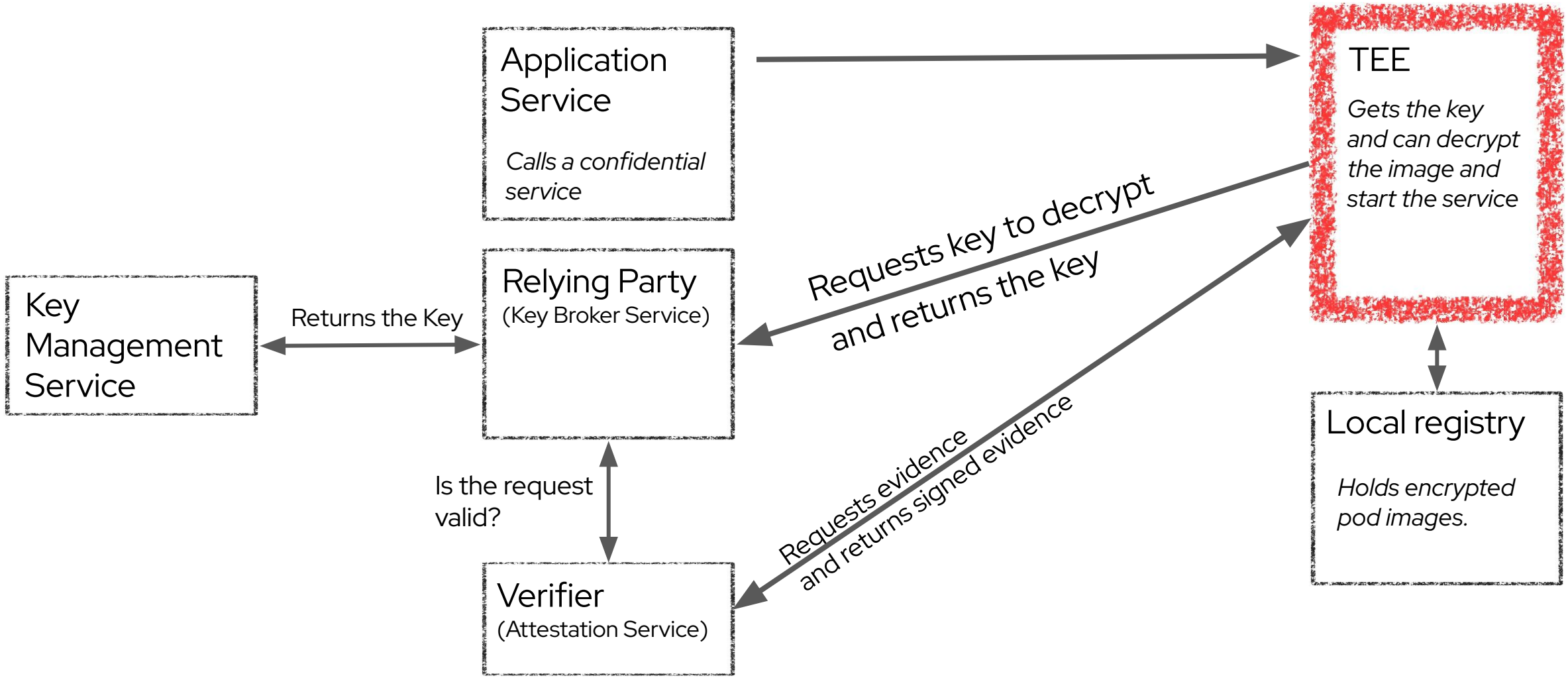
Flow of the attestation



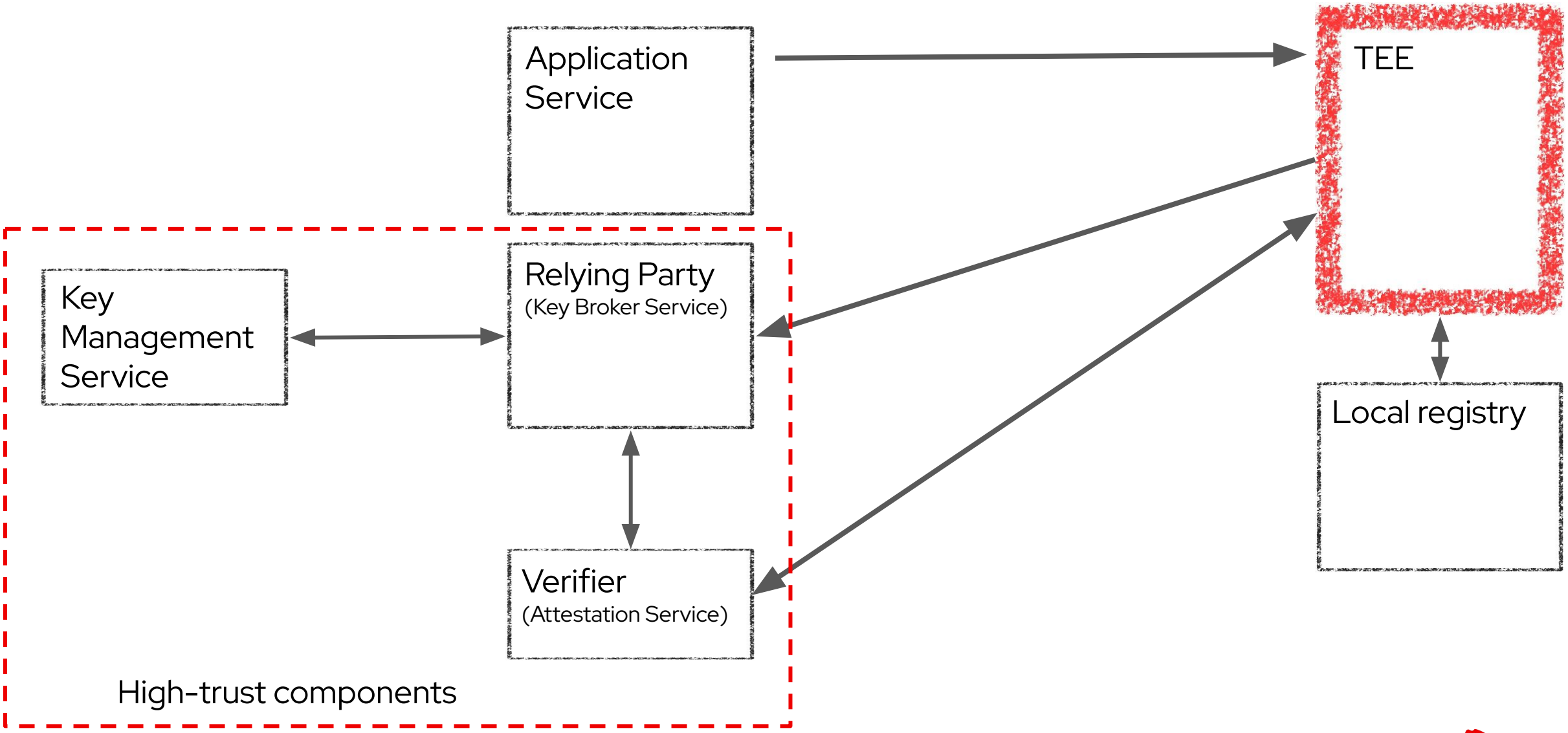
Flow of the attestation



Flow of the attestation

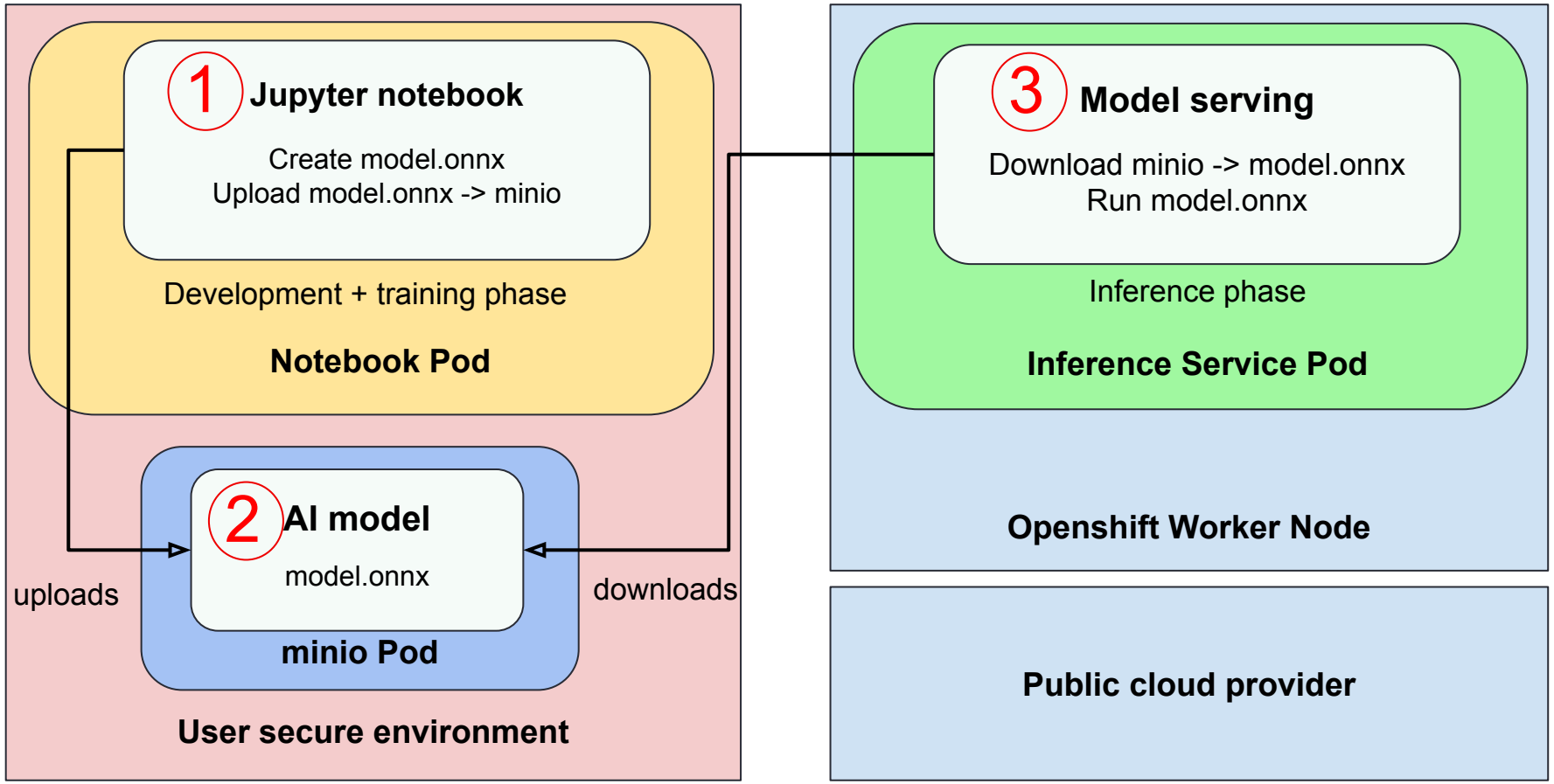


Flow of the attestation



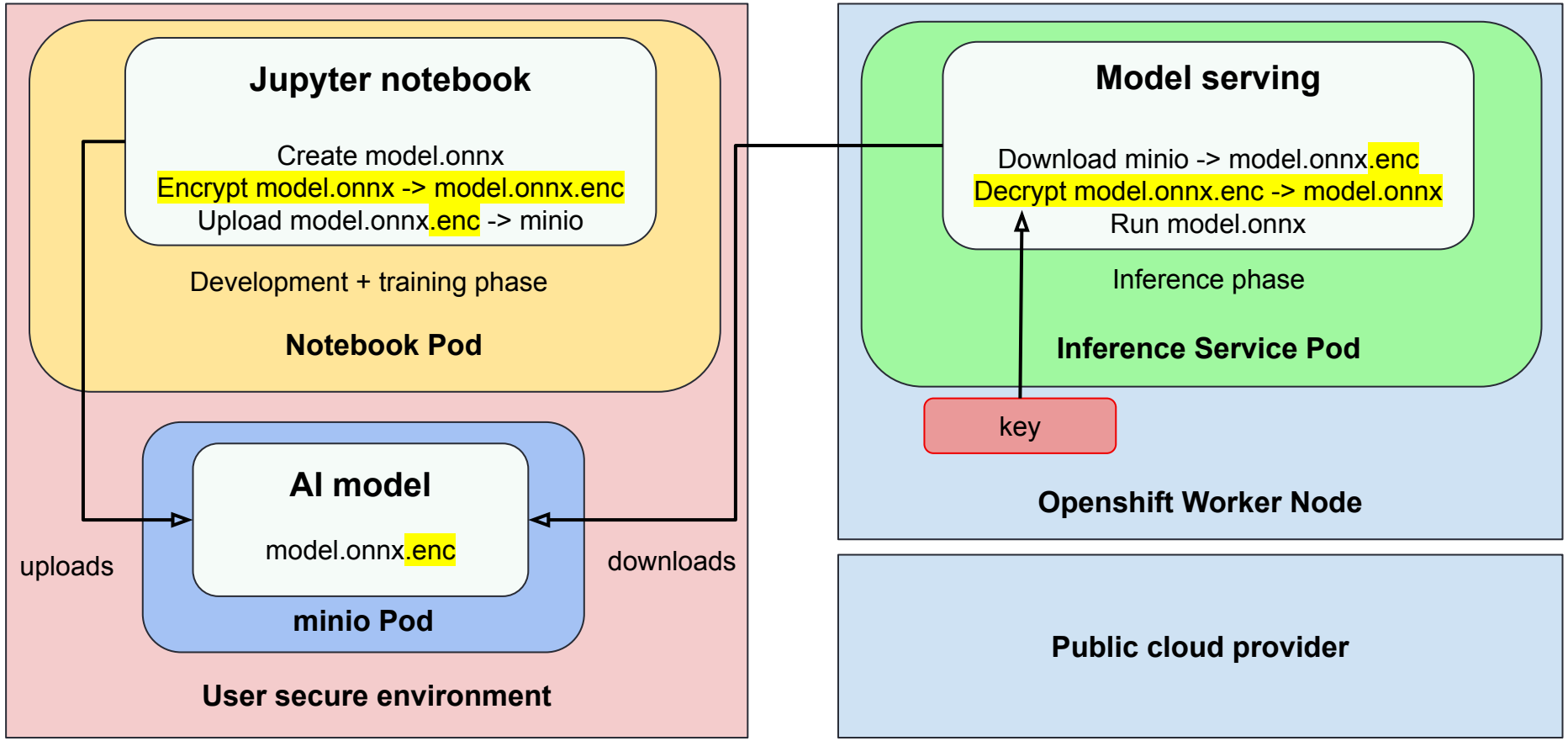
Red Hat

Assumptions



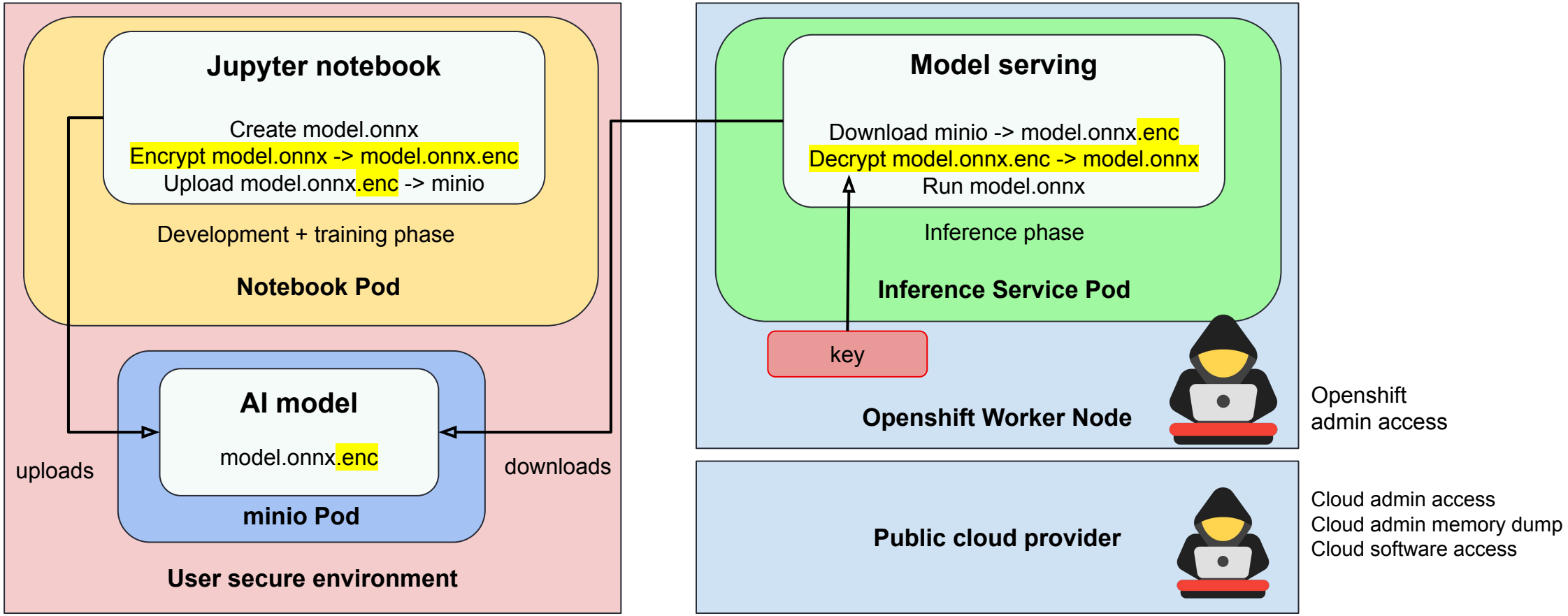
Red Hat

Fraud detection demo scenario with model encryption



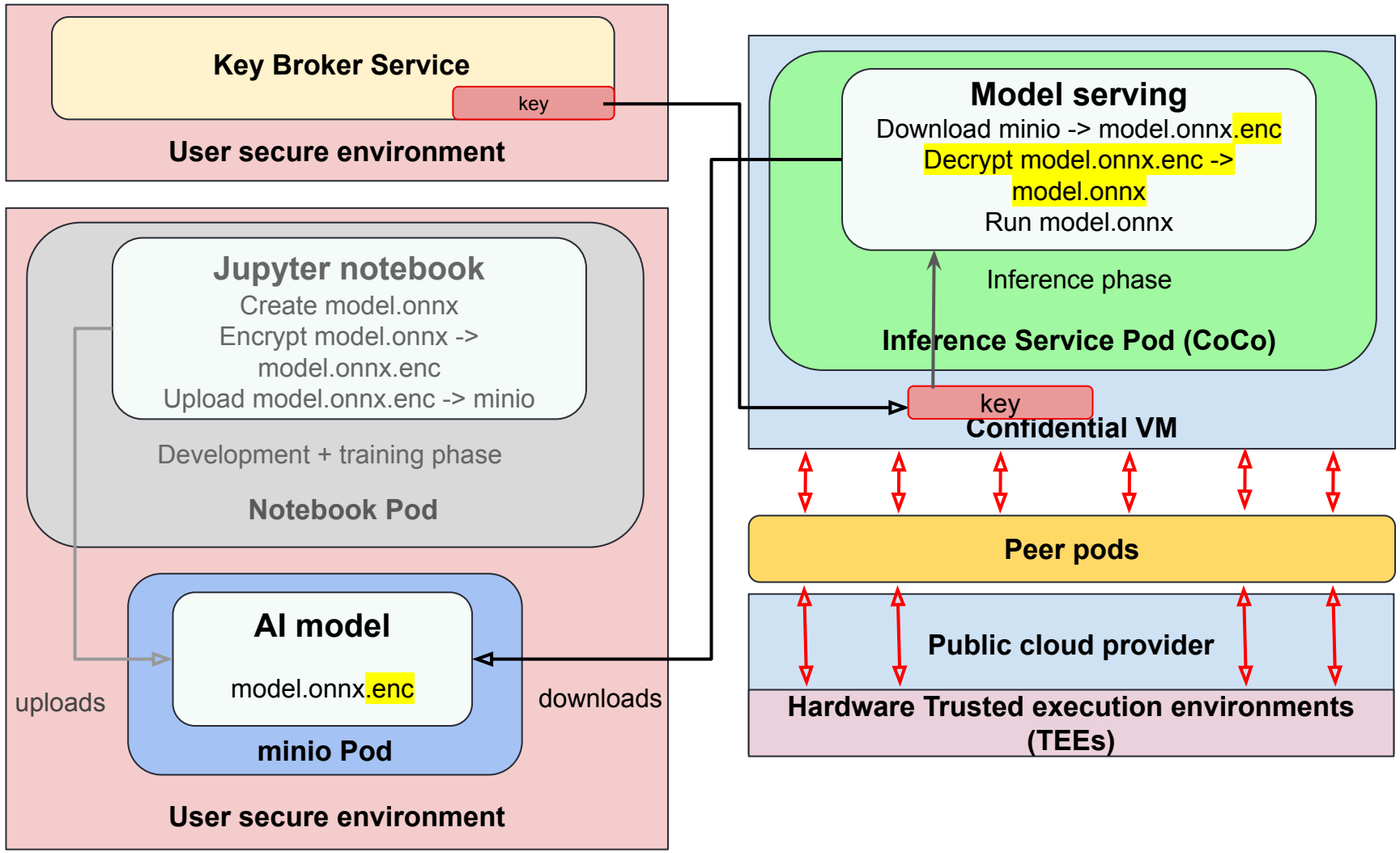
Red Hat

Fraud detection demo scenario with model encryption



Red Hat

Fraud detection demo scenario with CoCo



Red Hat

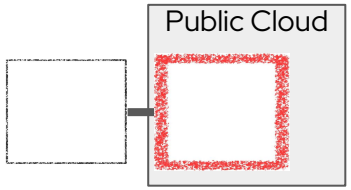
Categorization of Use cases

- What use cases are our customers telling us about?

Partner Interaction

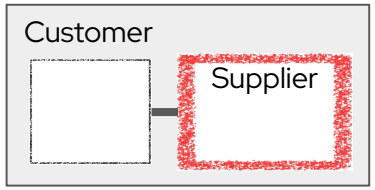
2 protected datasets interacting in confidential container

Secure Cloudburst



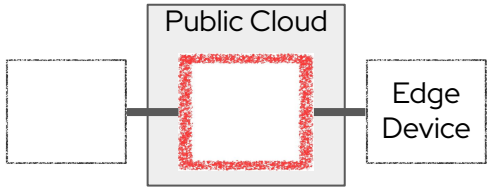
Using the public cloud to for peak workload or shared resources

IP Protection/Integrity



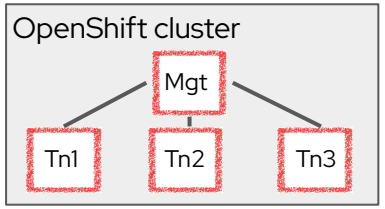
Protection of supplier data and business logic in customer environments

Edge use case



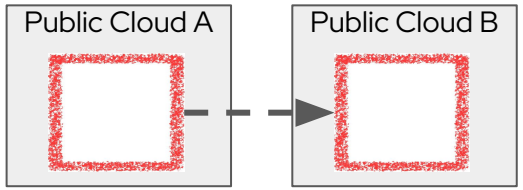
Protecting Edge device data in the public cloud for aggregation

Total Tenant Isolation



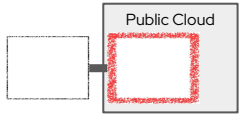
Isolating OpenShift Tenants

Digital Sovereignty



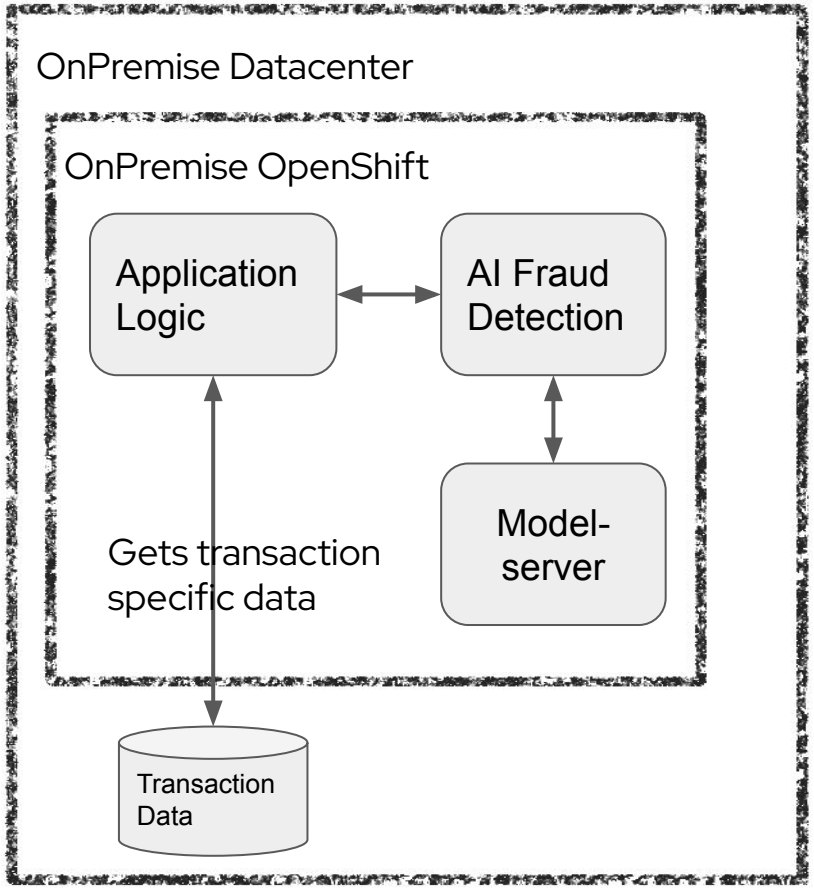
Encapsulating and moving workload from one provider to the next.

Red Hat

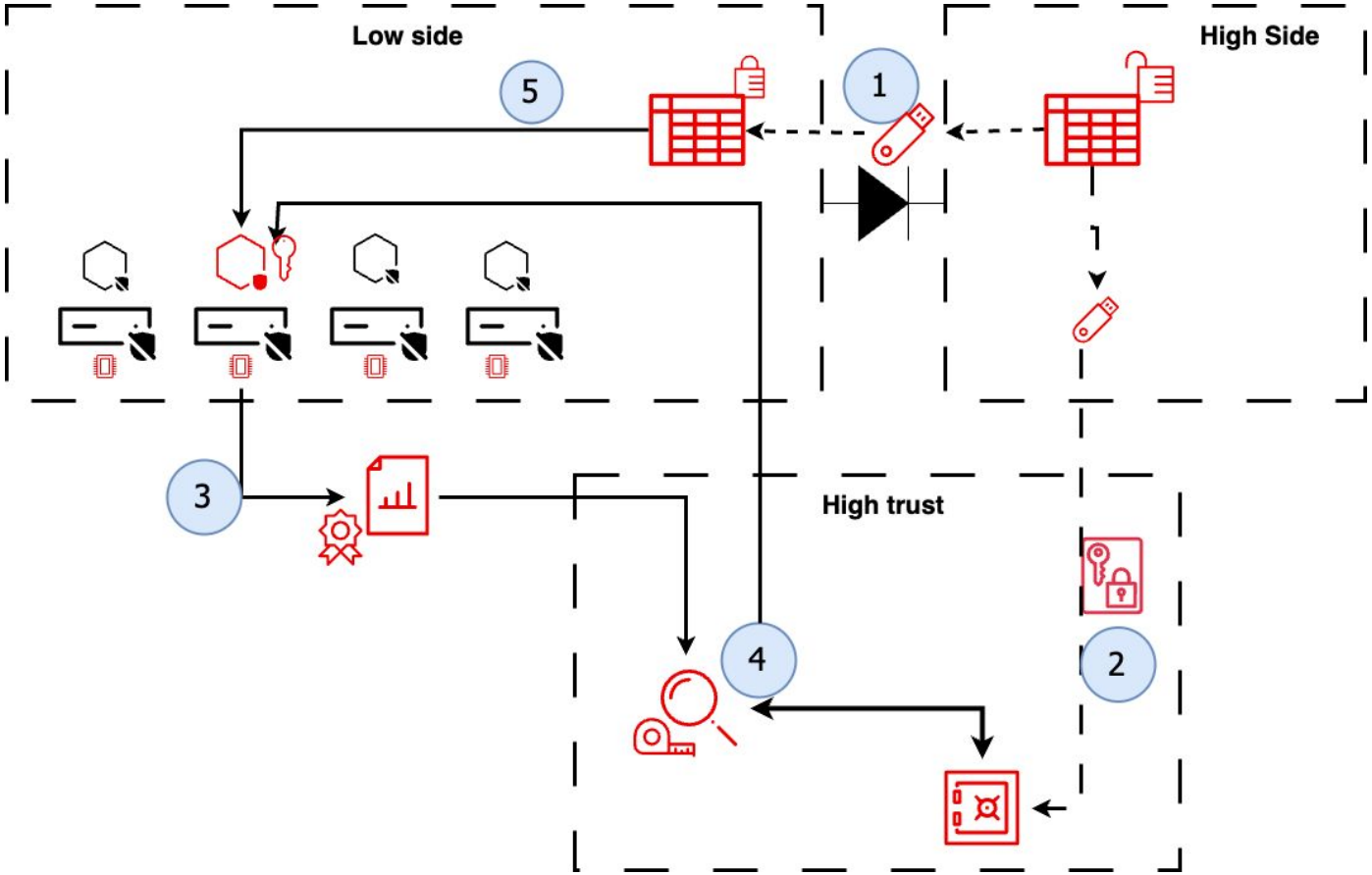


Confidential Computing Usecase (Fraud Analysis)

- Analyse transactions for fraud.
- AI model has been trained on premise.
- Transaction data and customer data is on premise.
- Goal is to use that AI model. Unfortunately company has not yet upgraded environment with GPUs.
- Usage of public cloud is difficult due to regulatory and GDPR requirements.



Confidential compute for ‘high to low’ computing



Red Hat

Next steps

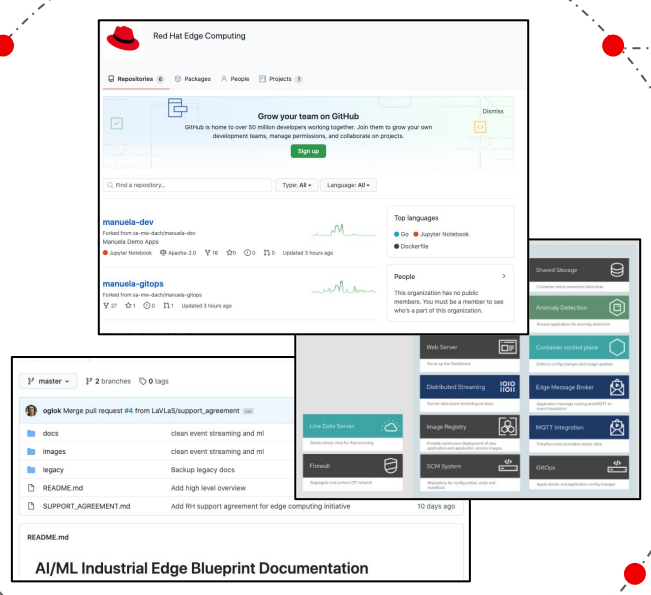
Validated Patterns : Simplifying the creation of use cases

Configuration as code

Go beyond documentation using GitOps process to simplify deployment

From POC to production

Ensure your teams are ready to operate at scale



Maintained over time

Each use case has a lifecycle to ensure they are kept up to date while they are being used

Tested as a use case

Confidence the configuration continues to work

Highly reproducible

So that you can scale out your deployments with consistency

Open for collaboration

Anyone can suggest improvements and contribute

Red Hat

FAQ

- How is the kata vm integrity protected?
  - Dm-verity is used to measure rootfs integrity
- What about oc exec / oc copy etc
  - oc exec is disabled by tenant side configuration (in the enclave) which is protected by the dm-verity device map
-

Key links

- OpenShift Sandbox Containers [GA]
- Confidential Containers [Upstream]
- Scripts used today
- Containers used today
- CoCo Blogs
  - What is the Confidential Containers project?
  - Understanding the Confidential Containers Attestation Flow
  - CoCo architecture
  - CoCo quick start guide
  - CoCo release notes

The End

Red Hat

Hardware implementations

- So this confidentiality needs to be implemented at the hardware level.
  - AMD SEV SNP: Confidentiality on a Core based VM and on the Memory

    https://www.amd.com/en/processors/epyc-confidential-computing-cloud

  - Intel SGX (core based) and TDX (VM)

    https://www.intel.com/content/www/us/en/products/docs/processors/xeon-accelerated/security-accelerators-product-brief.html

  - IBM z HyperProtect + Secure Execution

    HSM modules available to enable confidential computing environments
  - ARM CCA

    https://www.arm.com/architecture/security-features/arm-confidential-compute-architecture
  - AWS Nitro

    https://docs.aws.amazon.com/whitepapers/latest/security-design-of-aws-nitro-system
  - RISC-V (plan)

    https://github.com/riscv-non-isa/riscv-ap-tee/blob/main/specification/riscv-cove.pdf

Red Hat

Confidential Computing Organizations

- There are 2 standardizing organizations:
  - CCC (founded 2019)
    - The Confidential Computing Consortium (CCC) brings together hardware vendors, cloud providers, and software developers to accelerate the adoption of Trusted Execution Environment (TEE) technologies and standards.
    - https://confidentialcomputing.io/

You're in good company.

accenture · 蚂蚁集团 ANT GROUP · arm · Google · HUAWEI · intel. · Meta · Microsoft · Red Hat

  - CNCF (accepted 03/2022)
    - The Cloud Native Computing Foundation (CNCF) hosts critical components of the global technology infrastructure.
      https://www.cncf.io/

Red Hat

Red Hat (member)

Red Hat CNCF Platinum Member

CNCF Members · Platinum

Red Hat is a software company that offers enterprise open source software solutions.

Red Hat