Going Beyond Confidential Attestation with Trustee

Tobin Feldman-Fitzthum <tobin@ibm.com>
Claudio Carvalho <cclaudio@ibm.com>
Chris Porter <porter@ibm.com>
Niteesh Dubey <niteesh@us.ibm.com>
Daniele Buono <dbuono@us.ibm.com>

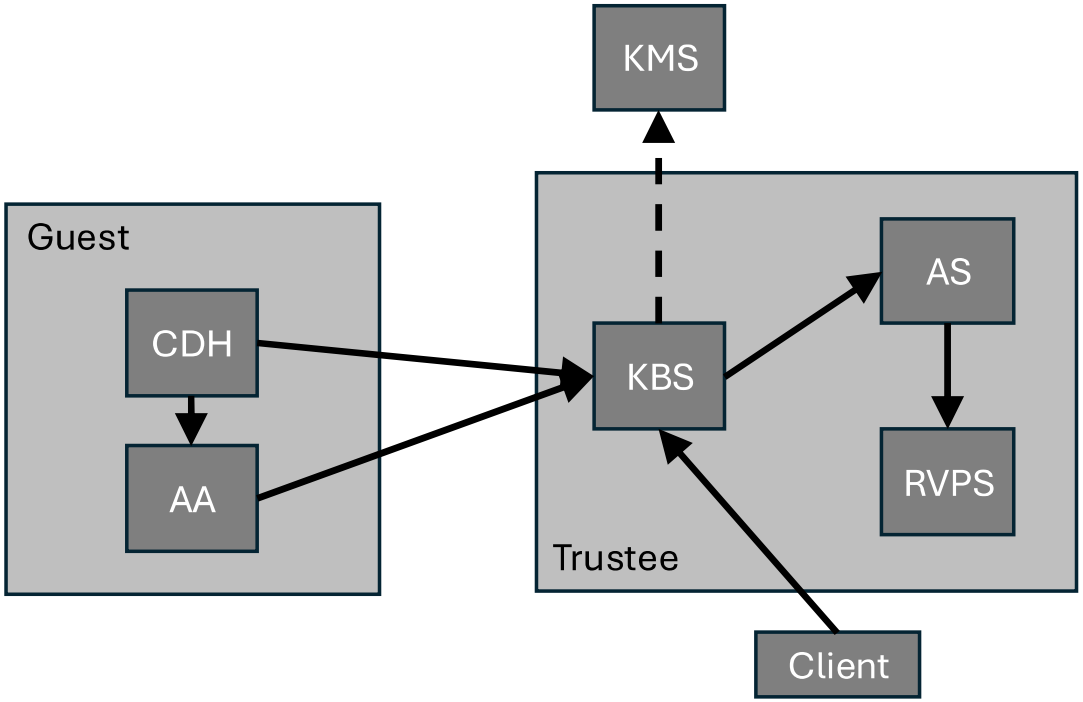
Confidential Computing Micro-Conference @ LPC24

Overview

- Re-introduction to Trustee
- EAT Integration Plan
- Runtime Attestation
- Device Attestation
- KBS Plugin and VPN

Trustee

- Formerly known as KBS
- Supports lots of platforms
  - SEV-SNP
  - TDX
  - SGX
  - CCA
  - TDX on Azure
  - SNP on Azure
  - IBM SE
  - CSV
- Supports generic workloads
- PR avg time to engagement: 5.5 hours



https://github.com/confidential-containers/trustee

KBS Protocol

Request

```
{
    /* KBS protocol version number used by KBC */
    "version": "0.1.1",
    /*
     * Type of HW-TEE platforms where KBC is located,
     * e.g. "intel-tdx", "amd-sev-snp", etc.
     */
    "tee": "$tee",
    /* Reserved fields to support some special requests sent by HW-TEE. */
    "extra-params": {}
}
```

Challenge

```
{
    /* Evidence freshness. */
    "nonce": "$nonce",
    /* Extra parameters to support some special HW-TEE attestation. */
    "extra-params": {}
}
```

Attestation

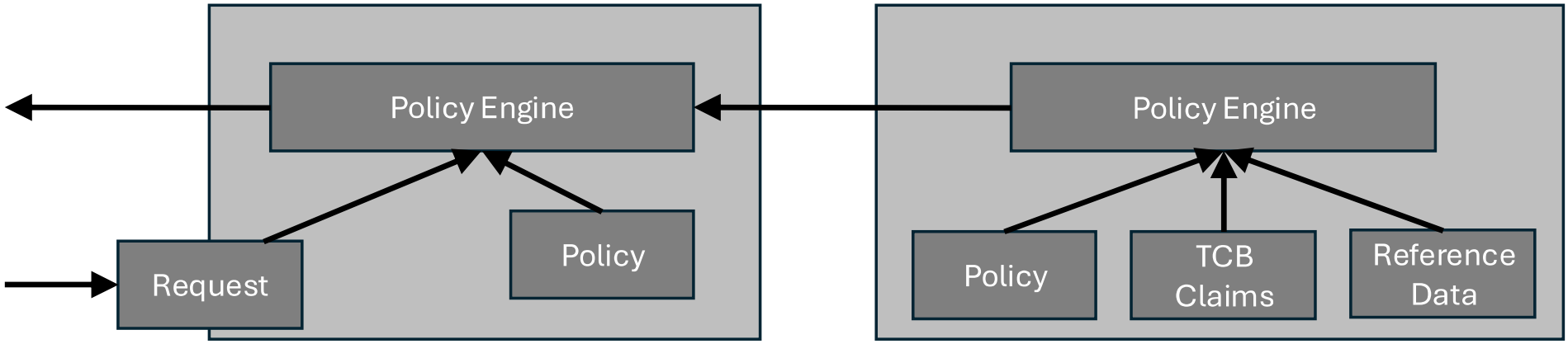
```
{
    /*
     * A JWK-formatted public key, generated by the KBC running in the HW-TEE.
     * It is valid until the next time an attestation is required. Its hash must
     * be included in the HW-TEE evidence and signed by the HW-TEE hardware.
     */
    "tee-pubkey": $pubkey

    /* The attestation evidence. Its format is specified by Attestation-Service. */
    "tee-evidence": {}
}
```

Response

```
{
    "protected": "$jose_header",
    "encrypted_key": "$encrypted_key",
    "iv": "$iv",
    "ciphertext": "$ciphertext",
    "tag": "$tag"
}
```

Dual Policy Model



KBS

Determines whether a resource will be released

Captures the workload configuration

Attestation Service

Determines whether TCB is valid

Captures the boot method of the guest

EAT Integration (Issue #353)

```
let token_claims = json!({
    "tee": to_variant_name(&tee)?,
     "evaluation-reports": policies,
      "tcb-status": flattened_claims,
      "reference-data": reference_data_map,
      "customized_claims": {
          "init_data": init_data_claims,
           "runtime_data": runtime_data_claims,
      },
});
```

Custom Attestation Claims (now)

```
pub struct Appraisal {
    /// The overall status of the appraisal represented by an AR4SI trustworthiness tier
    ///
    /// This is typically the lowest tier of all the claims that have been made (who's values have
    /// been set), though a verifier may chose to set it to a lower value.
    pub status: TrustTier,
    /// Contains the trustworthiness claims made in the appraisal
    pub trust_vector: TrustVector,
    /// Identifier of the policy applied by the verifier
    pub policy_id: Option<String>,
    /// Evidence claims extracted and annotated by the verifier from the evidence supplied by the
    /// attester
    pub annotated_evidence: BTreeMap<String, RawValue>,
    /// Addition claims made as part of the appraisal based on the policy indicated by `policy_id`
    pub policy_claims: BTreeMap<String, RawValue>,
    /// Claims about the public key that is being attested
    pub key_attestation: Option<KeyAttestation>,
}
```

EAT Appraisal

To do:

- Overhaul AS policies to produce an EAT (in Rego)
- Add some custom claims
  - TEE Platform
  - Reference Values

Runtime Attestation

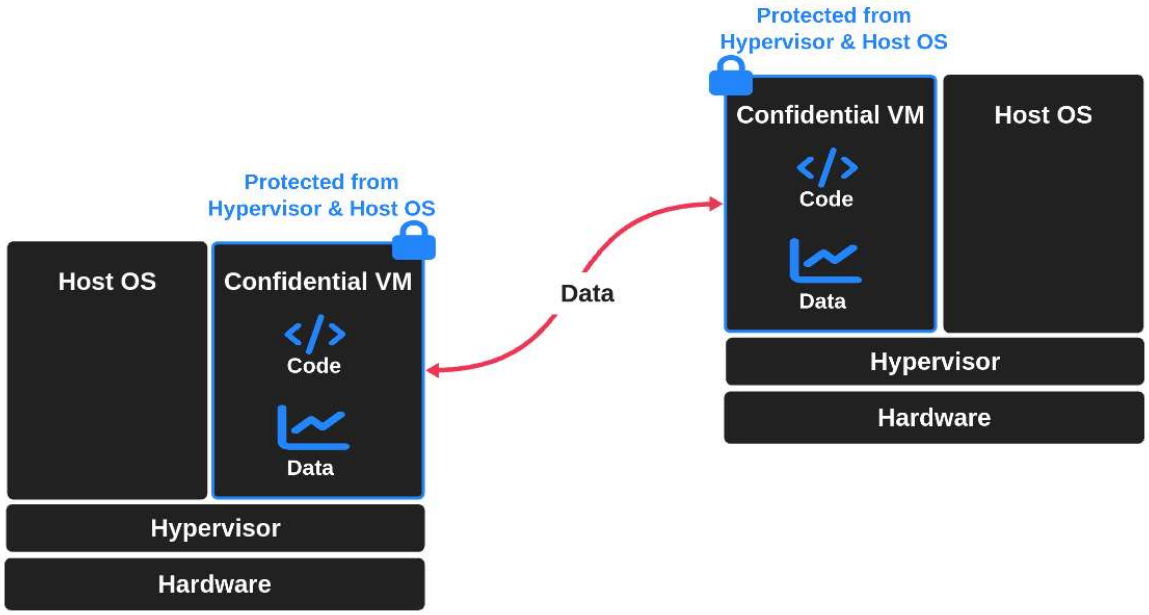
- The KBS protocol is executed lazily
  - The measurement is expected to be the same whenever the attester runs
  - The connection is valid until the attestation token expires

- You can use Trustee to inject TPM state
  - Runtime attestation and confidential attestation should be handled by two different, specialized, entities

Device Attestation

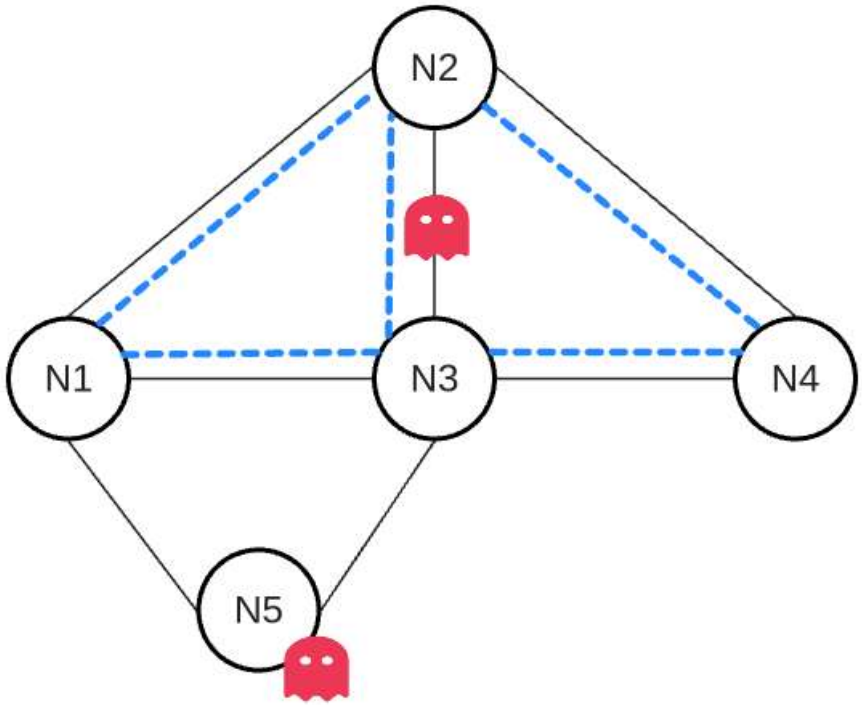
- KBS Protocol is executed lazily
  - Is this safe for device attestation?
  - Re-authentication seems promising
- Attesters/Verifiers do not need to be mutually exclusive
  - Can they share a challenge/nonce?
  - Fits nicely with multiple appraisals in EAT

PR#451 - Secure networking services



https://github.com/confidential-containers/trustee/pull/451

Protection with encrypted overlay network



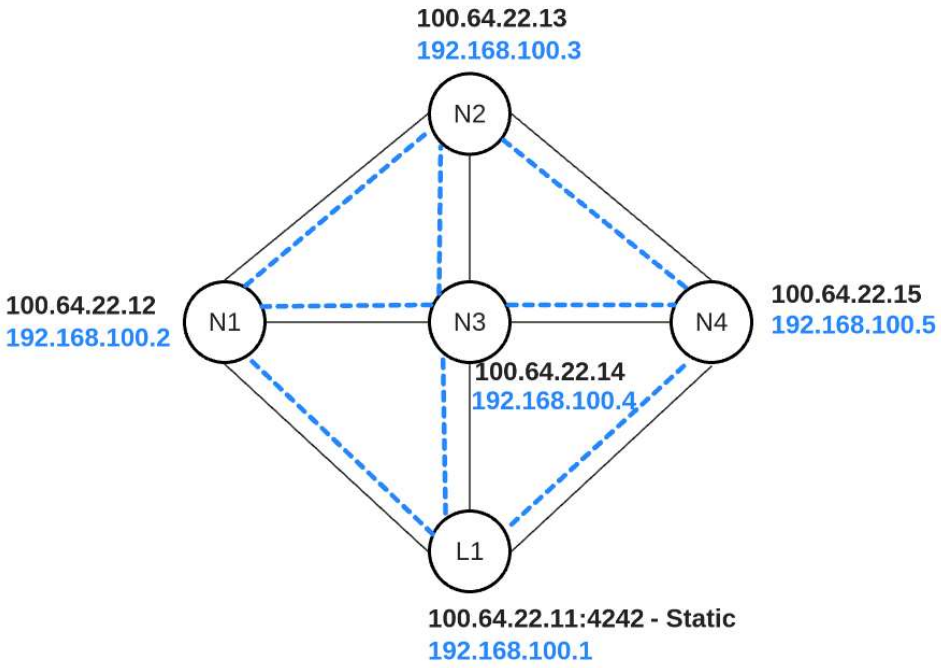
- Untrusted nodes:
  - Only authenticated nodes can join the overlay network
- Unencrypted links:
  - Traffic in the overlay network is encrypted

Nebula

- Provide a degree of off-the-shelf encrypted overlay support
- Designed to be fast, secure and scalable
  - Peer-to-peer, layer 3, virtual network
  - Supports TCP/UDP/ICMP traffic via TUN adapter with split-tunneling
- Connect nodes with on demand encrypted tunnels, without opening firewall ports

- https://nebula.defined.net/docs/
- https://github.com/slackhq/nebula
- Blog posts and presentations
- MIT License

Nebula overlay network



- Nebula releases
  - nebula-cert: generate keys, certs, CA and sign node certificates
  - nebula: runs node firewall and service
- Nebula CA
- Join at least one Lighthouse node, which helps nodes to discover routes to one another and assist with NAT traversal
- Join the nodes to the Lighthouse overlay network

PR#451- Plugin interface & nebula plugin

- get-resource(): retrieve resource from Trustee
  - ./kbs-client --url http://127.0.0.1:8080 get-resource --path "<repository>/<type>/<tag>"
- Plugin interface extends the get-resource interface
  - Dynamic resources
  - Additional parameters (Query String) to describe the resource
  - ./kbs-client --url http://127.0.0.1:8080 get resource --path "**plugin**/<plugname>/<resource><?arg1=v1&arg2=v2&...>"
- Nebula plugin
  - Nebula CA is created at Trustee start, if necessary
  - ./kbs-client --url http://127.0.0.1:8080 get-resource --path "plugin/nebula/credential?ip[ip]=10.9.8.2&ip[netbits]=21&name=node1"
  - Node credential is generated and returned **only if** the CVM is already attested
  - credential = {node_crt:[..], node_key: [...], ca_crt: [...]}

Discussion

- The Nebula Lighthouse provides DNS as experimental. How can we guarantee that traffic sent between two nodes within the Overlay Network always go through the nebula network interface?

- In a CVM, traffic encryption/decryption is a time expensive task. Can it be securely offloaded to the NIC? Isolation? Device attestation?