Confidential Computing in Computing Aware Network

draft-yang-teep-ccican-00
What is CAN

Computing-aware Networking (CAN), which is computing and network resource joint optimization based on the awareness, control and management over network and computing resources, to determine the appropriate service node, dispatch the service request and provide a better user experience.
What is CAN

Brief Architecture of CAN

Logic Layers of CAN

Operation Layer

Orchestration/Management Layer

Computing Resource
Consumer

Computing Resource
Provider

Computing Resource
Consumer

Computing Resource
Provider

Computing Resource
Provider

Computing Resource
Consumer

Difference between CAN and Cloud Computing:

CAN is based on network layer, computing task requirement and scheduling are carried by protocols in IP network. From the perspective of network layer, cloud computing is just application layer data which cannot be aware by network.

Advantage: Network latency awareness; Network topology awareness; Convergence with network resource and computing resource
Confidential Computing in CAN

Requirements for Confidential Computing

- If the computing resource is a third party source like the edge CR, which both the CAN and the network user cannot trust its security, then the Confidential Computing technique is needed.
- If the network user cannot trust the computing resource controlled by CAN like Central CR, then the Confidential Computing technique is needed.

Two components in CAN are mostly involved when using confidential computing:

- CPU of Computing Resource with confidential computing feature, like SGX, TDX, SEV, CCA, etc.
- O/M with confidential computing feature, like TEEP and RATs.
The confidential computing architecture in CAN based on TEEP and RATs working group.
Mapping between TEEP Concept and Instantiation:

<table>
<thead>
<tr>
<th>TEEP Concept</th>
<th>Instantiations</th>
</tr>
</thead>
<tbody>
<tr>
<td>M/OC, TAM and Middleware Repo</td>
<td>Computing Resource Orchestration in CAN</td>
</tr>
<tr>
<td>TEEP Broker</td>
<td>Function in OpenStack</td>
</tr>
<tr>
<td>TEEP Agent</td>
<td>OpenStack initiates guest VM, which includes TEEP Agent and specific Middleware like Enarx, Gramine, Occlum, etc.</td>
</tr>
<tr>
<td>Middleware</td>
<td></td>
</tr>
</tbody>
</table>
Step 1: App Owner asks for remote attestation.
Step 2: Middleware triggers Attesting Environment to generate remote attestation evidence for Targeting Environment.
Step 3: APP Owner gets the reference value, which the APP owner could generate with the source code and images, or could query from third party authority.
Step 4: Middleware return the evidence to App Owner. The Owner could match it with the reference value.

Mapping between RATs Concept and Instantiation:

<table>
<thead>
<tr>
<th>RATs Concept</th>
<th>Instantiations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attesting Environment</td>
<td>TDX/SGX: Quote Enclave, SEV: SP, CCA: RMM, etc.</td>
</tr>
<tr>
<td>Targeting Environment</td>
<td>Guest VM, which includes TEEP, Middleware and Application</td>
</tr>
</tbody>
</table>
Potential Next Step about confidential computing in network

- Specify TEEP use case about confidential computing.
- Specify RATs use case about confidential computing.
- A unified tool or specification for APP Owner to execute remote attestation and provisioning.
Thanks