OMEC Development and Deployment

- OMEC Repositories
- Deployment Use Case – Edge Gateway
- Development of the Edge Gateway
OMEC Network Function Repositories

Control Plane (CP)
- HSS
- PCRF
- CTF
- C3PO
- OpenMME

Data Plane (DP)
- CDR Router
- NGIC-RTC
- PDN(s)

LEGEND
- Currently available
- Currently available (& unidirectional)
- Work in progress

Single Frame (1 instance of each component)
- 40K Users
- 1K Control Plane TPS
- 42-80 CPU Cores
Deployment Use Case – Edge Gateway
Edge Gateway Use Case

• Description – Provide an EPC based SAEGW that select the closest data termination (Edge Site) for a user

• Constraints
  - Minimal impact to existing production systems
    - No upgrades to eNodeBs
    - No changes to Tracking Area configurations
  - Support dynamic and operator assigned edges

• Accomplished with
  - TAI DNS server
  - eNodeB-ID DNS server
  - topon – Colocation determination – standard process in 3GPP

• Solution maintains 3GPP compliance while supporting all scenarios
SGW-C DNS Queries

The SGW will resolve the responses to an IP Address
Scenarios

Scenario 1 - TAI with no Edge sites
Uses TAC Query Data (Business as usual)

Scenario 2 - TAI with single Edge site
Uses TAC Query Data

Scenario 3 - TAI with multiple Edge sites
Uses eNodeB Query Data

Simple logic: SGW-C queries both eNodeB and TAC.

If no eNodeB query answer is the response, TAC Query data is used. Scenarios 2 & 3 have a “fallback” to the Centralized Data Center.
SGW Query – More Detail

At the SAEGW-C

1. Receive S11 CSR with ULI including eNodeB-ID and APN.
2. Determine role of the gateway (SGW, SAEGW or PGW) for APN. (This determines the UPF interfaces we are looking for – Sxa and Sxb or merely Sxa)
3. Following TS 29.244 (Sx spec) for UPF Sx selection by both eNodeBID and TAI (this includes topological colocation if indicated per TS 29.303).
   1. eNodeB query goes to App DNS*.
   2. TAI query goes to existing iDNS*.
4. If no eNodeBID record is returned, keep this fact in memory. Otherwise use the eNodeB query data.
5. Select the UPF via the iDNS TAI data and keep processing.

* - This assumes that the DNS cache does not already have a valid query present.
Development to Support the Use Case
Current Development

- Focus is on NGIC-RTC and supporting repositories
- Deployment-VMs (our production systems use OpenStack and VMs) but use of Container is not a major concern
- New pattern and Construction Techniques development used

Function
- GTP-C
- PFCP
- Diameter
- All Stage 2 (TS 23.401) S11 GTP with GTP mobility procedures can be supported
- OAM
- Restoration and Recovery (TS 23.007)

Construction Techniques
- Stack Pattern
- Auto-generated protocol structures from the specifications directly
- State Machine Pattern
- OAM Patterns
NOTES
1 – Decode does NOT occur for FreeDiameter or fired Timers
2 - May not always be present, e.g. retransmission
Considerations: Would Multiple Diameter Applications ever result in multiple UNIX_FD socket in each direction?

1 – Message is not decoded but the Event is assigned

Cross Message (Request / Answer) checking can take place

Note: Decode is skipped in CP since FreeDiameter has already decoded
# Stack Layer - Pattern

<table>
<thead>
<tr>
<th>Services</th>
<th>GTP-C Layer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reliability of Message Delivery</td>
<td>Peer / Application</td>
</tr>
<tr>
<td>Endpoint Path Failure Detection</td>
<td>Peer</td>
</tr>
<tr>
<td>Piggyback messages</td>
<td>Peer / Application</td>
</tr>
<tr>
<td>Protocol Errors</td>
<td>Peer</td>
</tr>
<tr>
<td>Unsupported Versions</td>
<td>Peer</td>
</tr>
<tr>
<td>Message Invalid Length</td>
<td>Peer / Application</td>
</tr>
<tr>
<td>Unknown Message</td>
<td>Peer</td>
</tr>
<tr>
<td>Unexpected Message</td>
<td>Application</td>
</tr>
<tr>
<td>Missing Information Element</td>
<td>Application</td>
</tr>
<tr>
<td>Invalid Length Information Element</td>
<td>Peer / Application</td>
</tr>
<tr>
<td>Semantically Incorrect Information Element</td>
<td>Application</td>
</tr>
<tr>
<td>Unknown or Unexpected Information Element</td>
<td>Application</td>
</tr>
<tr>
<td>Repeated Information Elements</td>
<td>Application</td>
</tr>
<tr>
<td>Common Structure Error Handling</td>
<td>Application</td>
</tr>
<tr>
<td>Detection of Peer Reset</td>
<td>Peer (standard) / Application (possible)</td>
</tr>
</tbody>
</table>

**Layers**

- **Application**
- **Peer**
- **Transport**

1 - Protocol specific
Summary

Purpose - Provide an EPC based SAEGW that selects the closest data termination (Edge Site) for a user in NGIC-RTC

Approach - We’re working with design patterns and a common approach

Focus
• Edge Gateway Use case and impacted component
• Compliance to specifications
• Improving the way we develop and test the code for quick, efficient repeatability

Timeline - Code delivery this year but acceptance of delivery, testing and verification and readiness to release to OMEC will take time
How to Engage with Community

• On Github - https://github.com/omec-project

• Weekly Meetings - POC
  • TST - Oguz Sunay oguz@opennetworking.org
  • Architecture|Design|Engineering - Pingping Lin pingping@opennetworking.org
Thank You

Follow Up Links:
https://github.com/omec-project