Overload Control Data Analysis

(draft-campbell-dime-overload-data-analysis-00)

Ben Campbell
IETF86, Orlando

Current Mechanism Proposals

- Diameter overload Control Application (DOCA)
  - draft-korhonen-dime-ovl
  - Uses a dedicated Diameter application for overload reporting
- Mechanism for Diameter Overload Control (MDOC)
  - draft-roach-dime-overload-ctrl
  - Piggybacks overload reporting on existing messages
  - I made up the acronym ☺

Draft Purpose

- The draft attempts to analyze differences in the data models.
  - Goal to evolve to a common data model
- Draft does not attempt a general comparison
  - No conclusions here about relative merits

Fundamental Differences

- While the analysis focus on data elements, there are some some mechanisms differences that impact them
  - Non-Adjacent Nodes
    - MDOC as described is strictly hop-by-hop
    - DOCA may allow non-adjacent OC communication at some point, but doesn't address it I current revision
  - Scopes
    - MDOC has richer (and therefore more complex) idea of scopes

Fundamental Differences

- Stateless Mode
  - DOCA allows stateful and stateless modes
    - Nodes are not ***required*** to keep state, but may choose to do so in implementation-specific ways.
    - All parameters must be restated for each overload report
    - (Updated version seems to remove stateful mode?)
  - MDOC is always stateful.
- Soft State vs Hard State
  - MDOC always treats overload information as soft state
  - DOCA supports soft state, but treats overload as hard state in some circumstances

Naming Conventions

- Different naming styles
- MDOC prefixes things with “OC-”
  - e.g. OC-Scope
- DOCA uses “Overload” prefix for root level AVPs, and leaves grouped AVPs to context
  - e.g. Overload-Info, Supported-Scopes
- Not really important, but WG should pick a style.

Negotiating Capabilities

- Several data elements are used to negotiate capabilities at connection establishment
- These are in addition to the normal CCR/CCA usage to negotiate application support.
- When DOCA operates statelessly, negotiated parameters are hints
- MDOC holds negotiated values constant for the life of a connection.

Supported Scope Selection

- DOCA: OC-Scope – Bitmap of scopes supported by sender.
  - Defined values: "Host", "Realm", "Only Origin Realm"," Application Information", "Node Utilization Information", and "Application Priorities"
  - OC-Scope used both for declaration of supported scopes, and listing scopes for a given overload report.
  - DOCA overloads OC-Scope to include idicators that load information and priority may be included

Supported Scope Selection

- MDOC: Supported-Scopes
  - Defined Values: "Destination-Realm", "Application-ID", "Destination-Host", "Host", "Connection", "Session", and "Session-Group"
  - Separate parameters for declaring supported scopes, and listing scopes for an overload report.

Algorithm Selection

- DOCA: OC-Algorithm
  - Currently defined values: Drop, Throttle, Prioritize
  - Multiple values allowed. (What does it mean to combine them?)
- MDOC: Overload-Algorithm
  - Currently defined value: loss
  - Single value allowed for the life of a connection.

Application Selection

- DOCA: OC-Applications: Indicates applications of interest
- MDOC assumes overload reports apply to any and all applications crossing a connection.
  - Open Issue: Are there use cases for up front negotiation of applications of interest?

Report Frequency

- DOCA: OC-Tocl: Requested frequency of overload reports:
- MDOC: Piggy-backed on existing messages; rate of overload reports varies with rate of other messages.
  - Open Issue: Need further discussion about rate of overload reporting, regardless of the approach.

AVP Grouping

- DOCA: negotiation AVPs included at message root.
- MDOC: Load-Info: Grouped AVP acts as a container for other AVPs used in negotiation
  - Artifact of DOCA using a dedicated application vs MDOC piggybacking on existing messages.

Reporting Overload

- Several data elements are used for reporting of current load and overload information.
- Overload and load information is generally soft state for both mechanisms, but DOCA treats overload as hard state in some circumstances
- Since DOCA can operate statelessly, negotiated parameters are repeated in each overload report.

Report Scope

- DOCA: OC-Scope (same as for negotiation)
- MDOC: Load-Info-Scope – Octet stream with a type and value. Multiple values allowed.
  - DOCA does not include an explicit value, only a type. The value is inferred from context or other AVPs
    - e.g. MDOC allows you to say “realm: example.com”, while DOCA would say “realm: this realm”

Overload Severity

- DOCA:
  - OC-Level – Values 1-6 define discreet levels of increasing severity, with explicit guidance for each level.
  - OC-Sending-Rate: Indicates max sending rate for “throttle” algorithm
- MDOC: Overload-Metric – Abstract representation of load. Interpretation is algorithm specific.
  - for “loss” algorithm, a value of 1-100 to indicate requested percent of traffic reduction.
- Open Issue: abstract approach vs. fixed interpretation of AVPs?

Report Algorithm

- DOCA: OC_Algorithm. Multiple values allowed
- MDOC: n/a – algorithm selected during connection setup.

- Open Issues
  - Do we need to change the algorithm mid-connection?
  - What does it mean to have multiple algorithm values for the same report?

Report Expiration

- DOCA: Oc-Best-Before – Time of report expiration
- MDOC: Period-of-Validity – Number of seconds until report expiration

- Open Issue: Point in time vs time interval?

Current Load

- DOCA: OC-Utilization – Overall load (1-100)
- MDOC: Load – overall load (0-65535)
  - MDOC load range chosen to fit with the SRV weight field.
  - Open Issue: Which range?

Covered Applications

- DOCA: OC-Application – indicates Diameter applications of interest for a report
- MDOC: n/a
  - MDOC can use the application scope type to describe which applications a given report applies to.

Priority

- DOCA: OC-Priority – sets relative priority of applications listed in OC-Applications. May also be used to set the priority of a given message.
- MDOC: n/a
  - Relative priority between applications could be achieved by assigning different overload values to different application scopes
- Open Issue: Is OC-Priority just for the "Prioritize" algorithm?

Session Groups

- DOCA: n/a
- MDOC: Session-Group – allows a node to assign a session-group label to a session.
  - The node can later send a single overload report covering the entire group of sessions.
  - Useful for an agent that distributes sessions across servers, and one server fails or becomes overloaded.

Result Codes

- DOCA defines the following new result codes:
  - DIAMETER_NO_COMMON_SCOPE
  - DIAMETER_NO_COMMON_ALGORITHMS
  - DIAMETER_TOCL_TOO_BIG
  - DIAMETER_TOCL_TOO_SMALL
- MDOC defines DIAMETER_PEER_IN_OVERLOAD
  - MDOC has an MTI algorithms and MTI scopes, so failures to negotiate either are protocol violations
  - MDOC does not have the Tocl concept.
- Open Issue: Is DIAMETER_PEER_IN_OVERLOAD useful for both?

Where do we go from here?

- Does it make sense to create a common data model?
  - Are we likely to have more than one OC transport mechanism? Is one set of data elements likely to make sense for both?
  - Can we harmonize the different semantics?
- If so…
  - Should it be based on DOCA…
  - … MDOC …
  - … or something else?